

Using Human Computation to Acquire Novel Methods for Addressing Visual Analogy Problems on Intelligence Tests

David A. Joyner^{1,2}, Darren Bedwell¹, Chris Graham¹,
Warren Lemmon¹, Oscar Martinez¹, Ashok K. Goel¹

Design & Intelligence Laboratory¹
Georgia Institute of Technology
Atlanta, GA 30332 USA
{djoyner3, dbedwell3, cgraham36, wlemmon3, omartinez8}@gatech.edu; goel@cc.gatech.edu

Udacity²
2465 Latham Street
Mountain View, CA 94040 USA

Abstract

The Raven's Progressive Matrices (RPM) test is a commonly used test of intelligence. The literature suggests a variety of problem-solving methods for addressing RPM problems. For a graduate-level artificial intelligence class in Fall 2014, we asked students to develop intelligent agents that could address 123 RPM-inspired problems, essentially crowdsourcing RPM problem solving. The students in the class submitted 224 agents that used a wide variety of problem-solving methods. In this paper, we first report on the aggregate results of those 224 agents on the 123 problems, then focus specifically on four of the most creative, novel, and effective agents in the class. We find that the four agents, using four very different problem-solving methods, were all able to achieve significant success. This suggests the RPM test may be amenable to a wider range of problem-solving methods than previously reported. It also suggests that human computation might be an effective strategy for collecting a wide variety of methods for creative tasks.

Introduction

The Raven's Progressive Matrices (RPM) tests are a group of intelligence tests based on visual analogy problems (Raven, Raven, & Court 1998). In these problems, a matrix of visual frames is presented with a blank space; six or eight options are presented for filling in this space. Performance on RPM has been shown to correlate well with other intelligence tests (Snow, Kyllonen, & Marshalek 1984). Thus, although wholly visual, the RPM tests measure *general* human intelligence, and are often used as the psychometric measure of choice in educational and clinical settings.

Hunt (1974) suggested that humans use multiple problem-solving methods to address RPM problems, including "analytical" and "Gestalt" methods. Bringsjord & Schimanski (2003) have proposed intelligence tests such as RPM as a method of measuring the effectiveness of AI techniques. AI research has developed a variety of methods for addressing RPM and similar visual analogy problems,

including both "analytical" methods that typically use propositional representations (Evans 1968; Lovett, Forbus, & Usher 2009; O'Donoghue, Bohan & Keane 2006; Prade & Richard 2011; Ragni & Neubert 2014), and "Gestalt" methods that often use imagistic representations (Dastani, Induskhya & Scha 2003; Kunda, McGreggor, & Goel 2013; McGreggor & Goel 2014; Schewring et al. 2009). Another way of classifying the various methods is by control of processing. For example, some methods for addressing RPM problems, such as the affine method (Kunda, McGreggor & Goel 2013), first generate an answer based on the (partial) matrix, and test this answer by comparing it with each available choice; other methods, such as the fractal method (McGreggor, Kunda & Goel 2014), test each available answer by computing the degree of fit in the matrix. While it may appear that generation of answers is a necessary part of creativity, we posit that generating explanations for available answers is also creative.

The Raven's Test and Creativity

One major component in the value of the RPM test is its connection not only to intelligence, but also to creativity. Hunt (1974) laid the foundation for the creative nature of problem-solving methods on this test in identifying the two broad categories of methods mentioned previously, "Gestalt" and "analytical". Kirby & Lawson (1983) argued further that it is the diversity of problem-solving methods that makes the RPM test a valuable tool for assessing intelligence in humans. If creativity is in part the ability to develop novel, useful, and effective methods to a problem, then the RPM test's admission of multiple methods adds to its value as a tool for studying creative problem solving.

Second, Keating & Bobbitt (1998) argue that addressing many RPM problems requires metacognitive abilities to select among the available problem-solving methods, to monitor the progress of the selected method, to suspend or abandon the current method and move to a different method, and to combine insights from the use of multiple meth-

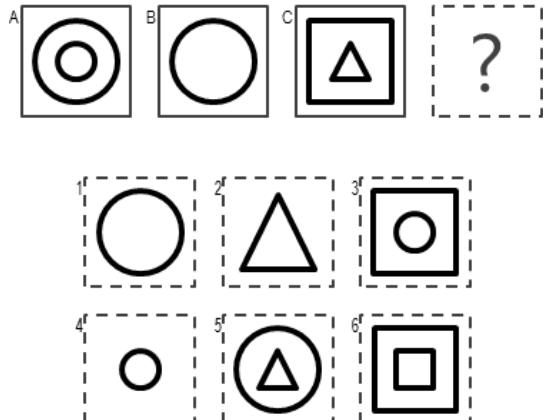


Figure 1: A 2x1 visual analogy problem. Although RPM tests do not have 2x1 problems, 20 2x1 problems are used as a soft introduction to solving visual analogies.

ods into one final answer choice. Third, the normatively correct choices for some RPM problems are often non-obvious, sometimes even unexpected, such as in the problem shown in Figure 1. Thus, from the perspective of both process (metacognitive processing) and product (unexpectedness of the answer), the RPM test measures not only intelligence, but also creativity.

One potential critique of the RPM test for studying creativity is that a set of answer choices are presented to the test-taker. However, this implies that the creative task necessarily entails generating a novel answer. The structure of the RPM problems turns this notion of creativity around: rather than generating an answer, the test-taker instead creatively generates an explanation for a particular answer choice. In Figure 1, for example, the most obvious answer would be a large square; however, none of the answer choices match this obvious answer. The presence of answer choices constrains the activity and forces the test-taker to creatively generate not an answer, but an explanation for why one of the presented choices is most compelling. This explanation is as much the output of the creativity process as the answer itself.

From the perspective of computational creativity, the above analysis makes the RPM test an excellent choice for designing, evaluating, and comparing new AI methods not only for intelligence, but also for creativity: the task admits a wide variety of AI methods characterized by different knowledge representations and different controls of processing. The question then becomes: how can we identify the novel techniques that may effectively address RPM problems?

We postulate that one strategy for acquiring new methods for addressing visual analogy problems on the RPM test is through crowdsourcing (Howe 2008), or, more accurately, human computation (Law & von Ahn 2011). Although crowdsourcing has typically been used for acquiring

domain knowledge, human computation also admits acquisition of problem-solving methods. Yet, it is also important to acquire new methods for addressing visual analogy problems not from any crowd, but from intelligent, educated, high-achieving humans who themselves are likely to do well on the RPM test.

The Experiment

In Fall 2014, we offered a new online Georgia Tech graduate-level CS 7637 course titled "CS 7637 Knowledge-Based AI: Cognitive Systems" as part of the new Georgia Tech Online MS in CS Program (Goel & Joyner 2014; Goel & Joyner 2015). We also offered an in-person class in parallel, with the two classes sharing the same syllabus and structure. The course describes its learning goals as, "to develop an understanding of (1) the basic architectures, representations and techniques for building knowledge-based AI agents, and (2) issues and methods of knowledge-based AI." Toward this end, students cover several knowledge representations (semantic networks, frames, scripts, formal logic), reasoning strategies (case-based reasoning, rule-based reasoning, model-base d reasoning), and target domains (computational creativity, design, metacognition). More comprehensive information on the structure and content of the class is available at the link above.

In previous offerings of the in-person class, we had used variants of problems on the RPM test to motivate the class projects (Goel, Kunda, Joyner, & Vattam 2013). Thus, we knew class projects based on the RPM test stimulated student engagement while providing an authentic opportunity to explore cutting-edge research. Therefore, in Fall of 2014, we again designed the class projects based on variants of problems on the RPM test. Students in both the online and in-person sections were asked to complete four projects that addressed 123 RPM-inspired problems in all, culminating in Project 4, wherein students designed agents that could answer all 123 problems using visual input. 224 students completed Project 4, addressing all the problems using the raw imagistic input. We collected all the data on these 224 Project 4 submissions, including the designs of the agents and their performance on the 123 problems.

In this paper, we will describe the results of this experiment. First, we will present at a high level the results of the 224 agents that were developed to address these RPM-inspired visual analogy problems. Second, we will examine in greater detail the design of four of the most creative and effective agents developed for the project. These agents operate according to four significantly different methods for reasoning about these problems. In describing these agents, we will clarify their relationship to elements of human creativity operationalized and instantiated in AI agents.

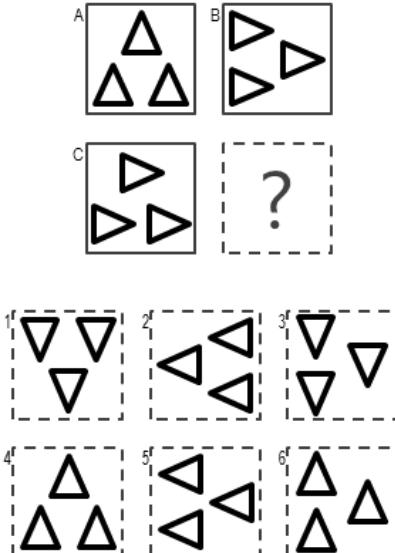


Figure 2: A 2x2 visual analogy problem, inspired by Raven's Progressive Matrices. In this paper, individual squares in a problem are called 'frames', while individual shapes within each frame are called 'objects'.

RPM-Inspired Visual Analogy Problems

The standard set of Raven's Progressive Matrices test is made of 60 visual analogy problems: 24 of the problems are 2x2 matrices, and 36 of the problems are 3x3 matrices. For copyright reasons, we have not yet been able to use actual RPM in these class projects. Instead, we have developed a set of 123 RPM-inspired problems. These problems are broken into three categories: 27 2x1 matrices (as shown in Figure 1), 48 2x2 matrices (as shown in Figure 2), and 48 3x3 matrices (as shown in Figure 3). Although there are no 2x1 matrices in the actual RPM test, these are included in our set to provide a simpler initial set of problems for students to address before moving on to more difficult problems.

To develop these RPM-inspired problems, we examined individual problems on the actual RPM tests (both the standard and the advanced test) and wrote problems to have a close correspondence with the problems on the actual tests. Although the individual shapes and their properties differ, these RPM-inspired problems mimic the same transformations and problem types as the actual standard and advanced RPM tests. These correspondences, however, only exist at the level of individual problems; not every RPM has a corresponding RPM-inspired problem in our problem sets, and some types of problems are present more often in our problem sets than in the actual RPM tests. Therefore, no claim is made that our RPM-inspired problem sets are equivalent to the RPM tests as a whole; we only claim that the individual problems capture the

same reasoning as problems on the original RPM tests. We are presently running two previously-designed agents (Kunda, McGregor & Goel 2011; McGregor, Kunda & Goel 2014) for solving the actual RPM tests against these new RPM-inspired problems in order to establish a conversion factor between the two sets.

The Projects

In the Fall 2014 version of the KBAI class, students completed a series of four projects. In the first three projects, students designed agents that could address 2x1, 2x2, and 3x3 matrix problems. During these projects, the input into these agents was propositional representations of the 123 RPM-inspired visual analogy problems. The propositional representations were written by the instructors of the course to prevent students from building inferential advantages into the representations. During the design of their agents, students could see 83 of these problems: the remaining 40 were designated 'Test' problems and were hidden from students in order to test their agents for generality. Thus, students were encouraged to construct agents with general problem-solving ability rather than agents that would tightly fit a small set of previously-seen problems.

By the end of project 3, students had completed an agent that could solve 2x1, 2x2, and 3x3 visual analogy problems based on propositional input. In project 4, students designed an agent that could solve these same problems using visual input. Here, students' agents read in the images directly from .PNG files, with one file representing each frame from the problem. Students' agents were run against the same 123 problems. Students' grades were dependent on performance on 100 of these problems (the remaining 23 were provided as challenge problems with no credit granted for correct answers), and 40 of these 100 problems were withheld as 'Test' problems. This paper focuses only on the agents designed in project 4, which took visual input.

Table 1: Performance on the eight sets of RPM-inspired problems (123 problems in all). "n" gives the number of problems in that set. "Avg." gives the average number of correct answers in that set for the 224 agents. "1", "2", "3", and "4" give the performance of the four agents described in further detail under 'Four Agents', below.

	<i>n</i>	Avg	1	2	3	4
2x1 Basic	20	8.8	18	14	17	12
2x1 Extra	7	1.5	4	1	7	2
2x2 Basic	20	8.8	18	16	20	14
2x2 Extra	8	2.5	7	4	7	7
2x2 Test	20	7.2	17	16	14	12
3x3 Basic	20	11.0	19	17	20	15
3x3 Extra	8	1.5	2	0	6	4
3x3 Test	20	7.9	16	15	11	13

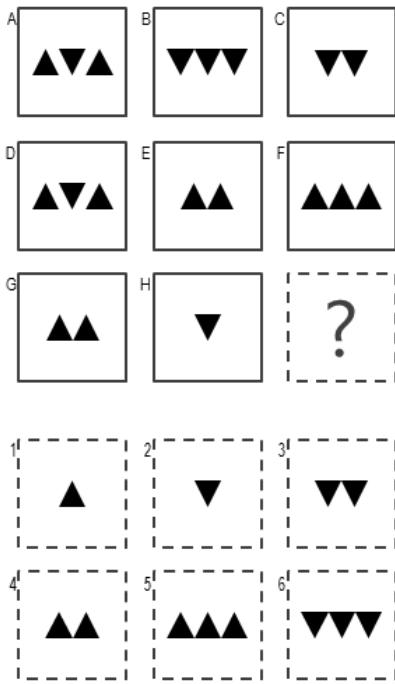


Figure 3: A 3x3 visual analogy problem, inspired by Raven's Progressive Matrices. Individual objects within frames in an RPM can be said to have ‘properties’; for example, some of the triangles in this problem have a 180° rotation as a property.

Aggregate Results

Students in the KBAI class submitted 224 agents, each of which ran against the 123 problems. The percentage of agents answering an individual problem correctly ranged from 87% (for the easiest 3x3 problem, which involved no transformations between frames) to 8% (for the hardest 3x3 problem, which demanded reasoning about the sum of the number of sides of multiple shapes). Among the problems completed for credit, one Test problem was correctly answered by only 10% of agents; this 2x2 problem involved two transformations – change-fill and remove-shape – that conflicted with one another.

Table 1 previously shows the performance of the agents as a whole, as well as the performance of the four agents highlighted below. The table is broken up by the eight distinct problem sets students addressed: ‘Basic’ sets were provided to students during the design of their agents and were evaluated for the project grade; ‘Test’ sets were not provided to students for the design of their agents and were evaluated for the project grade; ‘Extra’ sets were provided to students during the design of their agents but were not evaluated for the project grade. Agents’ scores on the Basic and Test sets comprised 70% of students’ project grades.

Perhaps surprisingly, students’ agents performed better on 3x3 problems than on 2x2 problems. While 3x3 problems allow more complex problem structures, such as

transformations in which two frames together determine the contents of a third, students noted that 3x3 problems gave their agents more information with which to work. With more information, their agents performed better, even on more complex problems.

Four Agents

After evaluating the aggregate results, we examined the problem-solving methods of several of the best-performing agents and identified a number of particularly novel and successful methods for addressing these RPM-inspired problems. The majority of the 224 submitted agents operated by first writing a propositional representation based on shape recognition, and then solving the problem propositionally; we describe the most successful agent using this method below, which combines contour recognition with problem classification. However, we also identified several other methods to solving these problems. Here, we describe three additional creative methods to solving RPM-inspired problems based on imagistic representations.

Agent 1: Contour Recognition & Reasoning

Agent 1 uses an intermediate propositional knowledge representation for working memory. In the agent’s representation, each frame in an RPM consists of objects, and each object consists of the following attributes: shape, size, fill, rotation, and relative-position to other shapes. A library of shapes was available to the agent, storing 20 basic shapes and features such as symmetry and corner count. Agent 1’s method has three phases: symbol extraction, top-down recognition, and bottom-up recognition.

Phase 1 uses image processing to extract a propositional representation for each problem. First, objects are found by isolating connected components, after which they are classified into shapes based on attributes of the object like corner count, edge lengths, and convexity. Other object attributes, including fill, rotation, size, and relative position are also computed in this phase.

Phase 2 uses top-down pattern finding. 19 pattern recognizers look for simple patterns that will be combined to form a pattern fingerprint. Recognizers include “constant rotation across objects in frame” (as seen between frames A and C in Figure 2) and “object count arithmetic sequence.” For each problem matrix, patterns are found and combined for all in-row, -column, and -diagonal relationships. The agent then chooses the answer with the largest set of matchers. In the event of a tie, Phase 3 begins.

Phase 3 performs bottom-up reasoning by splitting each problem into 2x1 sub-problems: 2 for 2x2 matrices and 29 for 3x3 matrices (including diagonal sub-problems). The agent solves each sub-problem, producing multiple answer choices, then uses majority-rule to make a final answer selection.

To solve a 2x1 sub-problem, (1) all object pairs from frame A to frame B are created; (2) all object pairs from frame C to the answer choices are created; (3) all mappings between object pairings from step one and step two are created; and (4) each mapping is given a score. The scoring function includes the intuitiveness of the transformation in step two and the strength of analogy in step three. For example, a mapping would be scored highly for intuition for mapping a triangle from frame A to a triangle in frame B. However, if a triangle in frame A instead mapped to a square in frame B, the best analogy would map triangles from frame C to squares in frame D. The highest scoring mapping is the most intuitive analogy. In the worst case, phase 3's runtime is $O((n!)^3)$, where n is object count per frame. To offset this, time limits were imposed.

To take the problem shown in Figure 3 as an example: during Phase 1, 31 shapes and 14 frames would be represented in a fashion similar to the following: frames: [{id: 1, objects:[{id: 1; shape: triangle; fill: yes; angle: 0; left-of: [2, 3]; size: medium},{id: 2; shape: triangle; fill: yes; angle: 180; left-of: [3]; size: medium}...]}, ...].

During Phase 2, each potential answer is inserted into the last cell of the matrix, and each pattern matcher runs. Here, the matcher labelled “remaining shapes after pairing” will match: each upright triangle in the first cell of a row or column is paired with a flipped version in the second cell, and the remaining triangles are checked to see if they match those of the third cell. Other matchers may also match the inserted choice, creating a more complex pattern. In the end, each potential answer will have a list of matchers associated with it, and the one with the longest list of matchers is selected. For this problem, the agent would choose the first answer choice. Because the problem would be solved in Phase 2, Phase 3 would not execute.

Agent 1 performed exceptionally well, correctly answering 101 of the 123 problems (88 of the 100 problems for credit). Agent 1’s general method of generating a representation based on prior shape knowledge also reflects the most common approach used in the class (as well as an approach used in prior literature, e.g. O’Donoghue, Bohan, & Keane 2006); however, Agent 1’s classification of multiple problem types goes beyond what the majority of agents attempt and plays a large role in its success.

Connecting with computational creativity, Agent 1 possesses the ability to creatively generate its own answers. Presently, Agent 1 operates by substituting each answer choice in the empty frame and evaluating its degree of fit to the problem’s transformations; however, implicit here is the idea of an ‘optimal’ fit for the remaining frame. Were the agent deprived of the answer choices, it could instead generate the optimal solution for the empty frame. Agent 1 is limited in this regard, however, in that it could only produce solutions that are comprised of the shapes in its shape library; Agent 1 cannot deal with novel shapes.

Agent 2: Shape-Agnostic Transformation Recognition

The second agent, Agent 2, operates in two stages. First, the agent detects and analyzes individual objects to produce a propositional representation, similar to Agent 1. The agent uses the individual properties to find relationships between objects in pairs of frames, and chooses the answer that best fits the relationships that are found. Agent 2’s high-level process thus resembles Agent 1’s in its initial phase of translating imagistic representations into propositional ones; however, it differs in that it does not rely on prior shape knowledge. Agent 2 derives the structure and content of the problem from within the problem, rather than based on prior knowledge of shapes and features.

The agent begins by recording visual measurements for each object in the problem and using a simple clustering method to partition similar objects into shape groups. The agent records the width/height ratio of an object and the amount of whitespace “outside” of the object’s boundaries in its cropped region. Without predefined knowledge of triangles and squares, the agent instead categorizes shapes based on these properties and gives them arbitrary names. For example, the agent may label all triangles as “shape1” and all squares as “shape2”, even if the individual objects vary in size and other properties across the problem, based on these measurements. To account for variations in the measurements, objects are rotated to optimize an arbitrary scoring function. This also helps determine relative rotation angles between objects which are necessary in certain problems.

To take an example, in Figure 1, there are no overlapping objects in the frames. Individual objects are easily isolated, and the shapes of these objects are distinguished by the relative outside whitespace. Other properties, such as relative size and position, are also computed. In frames A and B, the agent records as the target relationship that the single object in frame B has the same shape (shape2) as both of the objects in frame A and the same size as the larger object in frame A. The agent then compares frame C with each answer frame to find the closest match to this relationship. An exact match is not possible because frame C contains two different shapes (shape1 and shape3) rather than a single shape. The correct answer, frame 2 with the large triangle (shape3), is chosen because it matches all aspects of the target relationship other than the object matching the shape of the smaller object. Thus, the concept of shape is used to mark objects as being different from or similar to other objects, and as long as the agent correctly observes those differences in the visual analysis portion it will have enough information to solve the problem.

The process for the problems in Figures 2 and 3 is similar, although the addition of rotating objects demands the

agent's rotation logic. For example, in the first frame of Figure 3, the two outer triangles are already at the "ideal" rotation angle and are given an angle value of 0 degrees, whereas the middle triangle would reach the same "ideal" value after being rotated 180 degrees. As noted before, the primary difference between Agent 1 and Agent 2 is that while Agent 1 relies on prior knowledge of shapes and their potential properties, Agent 2 takes a grounded method to identifying shapes in a frame. Thus, while Agent 1 will fail to recognize previously unseen shapes, Agent 2 is equipped to address previously unidentified shapes.

Agent 2 performed exceptionally well, correctly answering 83 of the 123 problems (78 of the 100 problems for credit). It is notable, though, that Agent 2's performance lagged behind on the 'Extra' problem sets; many of these sets included transformations, such as counting the sides of a shape, for which Agent 2's more visually-oriented method does not account. We also hypothesize Agent 2 would show greater success on problems featuring previously unseen shapes that humans could similarly address, but no such problems were included here.

Like Agent 1, Agent 2 can also generate novel answers rather than select them from a set of possible answers. The paragraph above acknowledged that on the problem presented in Figure 1, the most-obvious answer to Agent 2 is not present among the answer candidates. To have a 'most obvious' answer prior to examining the choices, Agent 2 must generate its own solutions. This also reveals how the presence of candidate answers can encourage creativity by introducing new constraints. It is creative to generate novel solutions from scratch, but it is also creative to generate arguments for available non-obvious solutions.

Agent 3: Visual Heuristics

In contrast to Agents 1 and 2, Agent 3 does not derive any representation of the visual analogy problems. Agent 3 begins from the supposition that it is fundamental to reduce the input space to something both *manageable* and *meaningful* for the agent to be able to *compute* and *correctly guess* an answer from the given choices. Agents 1 and 2 do so by reducing the input space to a propositional representation; Agent 3 reduces the input space to sets of contiguous non-white pixels.

Agent 3 takes each possible answer choice and computes the likelihood it is correct. To do so, the agent takes a series of measurements capturing the relationship between each training pair, which is described by any two adjacent cells in the matrix. It then compares those measurements against each of the test-answer pairs, the combinations of any cell adjacent to the empty slot and each answer choice. Each comparison, if significant enough, casts a vote for the current answer as the likely answer with a weight directly proportional to the believed similarity of the cells. The most-voted answer is selected as the agent's answer.

Many relationship measurements were evaluated, such as grid-based similarity, histogram-based similarity, and affine transformations. After multiple iterations, few measures were needed to yield the best performance. In the final design, the agent only uses the following two measurements:

- **Dark pixel ratio:** the difference in percentage of the number of dark-colored pixels with respect to the total number of pixels in the contiguous pixel sets of two matrix cells.
- **Intersection pixel ratio:** the difference in percentage of the number of dark-colored pixels present at the same coordinates with respect to the total number of dark-colored pixels in both matrix cells for a given set of contiguous pixels.

For example, in Figure 1, the intersection pixel ratio would lead the agent to vote for the answers containing an outer square; this is analogous to the most logical answer to the problem, an outer square with the inner object removed. Counterintuitively, the correct answer is just the expanded triangle, but the agent would also vote for that answer based on the dark pixel ratio's similarity to the most logical answer. Hence, thanks to the simple metrics used, the agent is "immune" to problems that may appear deceiving at first glance or may involve convoluted transformations. Although for this particular example, the agent picked answer 6, the correct answer was evaluated to be only 6.76% less likely to be correct.

Agent 3 performed exceptionally well, correctly answering 102 of the 123 problems (82 of the 100 problems for credit). Agent 3 gave the most correct answers of any agent, although a greater proportion of its correct answers were previously-seen problems than Agent 1's similarly high performance. This may suggest that the iterations examining the effectiveness of multiple measures of similarity may have overfit the agent's reasoning to those problems, and that further development with more problems may expand the set of desirable measurements.

Unlike Agents 1 and 2, Agent 3 does not have the capability of generating an answer choice rather than selecting from a set of presented answer choices. This is because while Agents 1 and 2 operate under an implicit ranking of possible choices culminating in an ideal choice, Agent 3 might find numerous options equally ideal, and thus could generate thousands of candidate selections.

Agent 4: Hybrid Reasoning

Agents 1 and 2 use propositional representations of the target problem while Agent 3 uses purely imagistic representations; Agent 4, by contrast, leverages both and takes a hybrid method. This method asks the question: can an agent quickly find patterns and relationships in a problem through a high-level visual comparison? If the agent can find high-level visual relationships quickly, it can efficient-

ly formulate a solution without any further propositional understanding of the problem. If no such visual relationships are found, the agent may look for lower level propositional relationships present in the problem.

Thus, Agent 4 starts by examining frames for visual relationships and transformations that can be quickly detected by visual inspection. The agent uses image similarity to detect rotation, vertical and horizontal reflection, the identity transformation, image addition, XOR, and NOR. If this process detects the presence of one of these relationships within a matrix problem, the agent generates a prospective solution and looks for a matching answer. For example, in Figure 1, the transformation between frame A and frame B would be identified through the XOR transformation, which searches for pixels present in only one of two frames. Similarly, in Figure 2, the transformation between frame A and frame B would be identified through the rotation transformation; the agent would (successfully) identify frame 3 as a frame that would complete the same rotation transformation when paired with frame C.

This imagistic method was successful in finding solutions to over 20% of the problems, and it was much more computationally efficient compared to extracting propositional representations from the images; this is notable in that it acknowledges the different levels of effort applied by humans in solving these problems. Results could be further improved by searching for more types of high-level relationships and transformations, by applying transformations at a lower granularity than at the image level, and by improving the image comparison. For example, at present, Agent 4 is unable to detect the visual transformations between parts of frames in Figure 2.

This visual method has difficulty finding relationships that cannot be represented through affine transformations, such as problems involving prior knowledge of shapes and properties represented in the frames. When the agent is confronted with problems like these, it will try to find low-level relationships using contour recognition to identify shapes and object properties, ultimately leading to a method similar to Agent 1.

Agent 4 performed exceptionally well, correctly answering 79 of the 123 problems (66 of the 100 problems for credit). Although these scores are the lowest among these four agents, they are in the top 10% of agents submitted. Moreover, Agent 4 may represent the best approximation of human reasoning; humans can discuss problems in both visual and propositional terms (Kunda, McGregor & Goel 2011), and Agent 4 similarly can do both.

As noted in the description above, during the first phase of its reasoning, Agent 4 generates prospective solutions and compares those prospective solutions to the answer choices. Thus, it already engages in creative answer generation and compares the generated answers to the candidate solutions.

Discussion

Agents 1 and 2 above exemplify Hunt's (1974) analytical, propositional reasoning strategies for addressing RPM problems. Agent 1 extracts propositional representations that describe the shapes, spatial relations, and transformations from the input images, and then operates on those representations. Agent 2 also extracts propositional representations, but these representations are grounded in the transformations between objects: it has no prior knowledge of shapes, but rather the ability to generate representations of the transformations themselves. Agents 3, on the other hand, exemplifies Hunt's "Gestalt" visual reasoning strategy for RPM. It uses visual abstractions over problems to approximate the answer even without precise knowledge of the transformations between frames. Agent 4 combines the two methods: it first leverages the immediately-identifiable "intuitive" answer that can be established from accessible visual transformations before resorting to more complex propositional reasoning strategies. Thus, Agent 4 demonstrates the possibility of creatively combining methods. As far as we know, the precise strategies used by these agents have not appeared in the literature on the RPM test.

These four agents, along with the 220 other agents developed over the course of this project, reflect the ability of AI agents to succeed on a test of human intelligence that relies on creative and flexible problem-solving. This experiment suggests that there may be no one single "right" problem-solving strategy for the RPM test, that creativity on the RPM test may entail a large number of problem-solving strategies, and that we have so far discovered only a subset of creative problem-solving strategies. Future research along these same lines will test future agents against the authentic RPM test; examine patterns of errors in agents' performance for comparison to human performance (Kunda et al. 2013) including atypical cognition (Kunda & Goel 2011); and better articulate the strengths and weaknesses of different methods (Lynn, Allik, & Irving 2004; Kunda et al. 2013). We will also examine merging multiple agents into a single agent equipped with metacognitive ability to select among the different strategies, thus more closely approximating factors that determine human success on such tests (Keating & Bobbitt 1978).

Conclusions

The RPM test admits many problem-solving methods, which in part is what makes it a good test of intelligence and creativity. The various problem-solving methods differ in both the knowledge representations and control of processing they use. In this paper we described a human computation strategy for acquiring novel problem-solving methods for addressing RPM-inspired visual analogy problems. This strategy resulted in the design of 224 AI agents for addressing 123 visual analogy problems. Some of the

agent designs were both novel and effective: we described four of these agent designs.

An important issue in computational creativity is how to acquire knowledge of creative methods. Our research suggests that human computation may be a useful strategy for this acquisition, especially when the computation comes from intelligent, educated, high-achieving humans who themselves are likely to do well on a creative task.

Acknowledgements

We thank all 224 students in both the in-person and online sections of CS 7637 KBAI course at Georgia Tech in Fall 2014. Goel was the primary instructor of both sections; Joyner was the course developer and head TA of the online section; Lemmon, Graham, Martinez, and Bedwell were four students in the online course and developed agents 1, 2, 3, and 4, respectively.

We are grateful to Maithilee Kunda and Keith McGregor for their prior work on which this project builds. We also thank the course's teaching team: Lianghao Chen, Amish Goyal, Xuan Jiang, Sridevi Koushik, Rishikesh Kulkarni, Rochelle Lobo, Shailesh Lohia, Nilesh More, and Sriya Sarathy. We also thank the anonymous reviewers of this paper: their comments truly helped improve the discussion.

References

- Bringsjord, S., & Schimanski, B. (2003). What is Artificial Intelligence? Psychometric AI as an answer. In *Procs. 18th IJCAI*, 887-893.
- Dastani, M., Indurkhy, B., & Scha, R. (2003). Analogical Perception in Pattern Completion. *JETAI* 15(4), 489-511.
- Evans, T. (1967). A Program for the Solution of a Class of Geometric Analogy Intelligence-Test Questions. In M. Minsky (ed.) *Semantic Information Processing*. MIT Press.
- Goel, A. & Joyner, D. (2014). CS7637: Knowledge-Based AI: Cognitive Systems [Online Course]. Retrieved from <http://www.omscs.gatech.edu/cs-7637-knowledge-based-artificial-intelligence-cognitive-systems/>
- Goel, A. & Joyner, D. (2015). An Experiment in Teaching Cognitive Systems Online. Technical Report, Georgia Institute of Technology.
- Goel, A., Kunda, M., Joyner, D., & Vattam, S. (2013). Learning about Representational Modality: Design and Programming Projects for Knowledge-Based AI. In *Fourth AAAI Symposium on Educational Advances in Artificial Intelligence*.
- Howe, J. (2008). *Crowdsourcing: Why the Power of the Crowd is Driving the Future of Business*. Crown.
- Hunt, E. (1974). Quote the raven? Nevermore! In L. W. Gregg (Ed.), *Knowledge and Cognition*. 129-158. Hillsdale, NJ: Erlbaum.
- Keating, D. , & Bobbitt, B. (1978). Individual and developmental differences in cognitive-processing components of mental ability. *Child Development*, 155-167.
- Kirby, J., & Lawson, M. (1983). Effects of strategy training on progressive matrices performance. *Contemporary Educational Psychology*, 8(2), 127-140.
- Kunda, M., & Goel, A. (2011). Thinking in Pictures as a Cognitive Account of Autism. *Journal of Autism and Developmental Disorders*, 41(9), 1157-1177.
- Kunda, M., McGregor, K., & Goel, A. (2013). A Computational Model for Solving Problems from the Raven's Progressive Matrices Intelligence test using Iconic Visual Representations. *Cognitive Systems Research*, 22, 47-66.
- Kunda, M., Soulieres, I., Rozga, A., & Goel, A. (2013). Methods for Classifying Errors on the Raven's Standard Progressive Matrices Test. In *Proceedings of the 35th Annual Meeting of the Cognitive Science Society*, 2796-2801. Berlin, Germany.
- Law, E., & von Ahn, L. (2011). *Human Computation*. Morgan & Claypool.
- Lovett, A., Tomai, E., Forbus, K. & Usher, J. (2009). Solving geometric analogy problems through two-stage analogical mapping. *Cognitive Science* 33(7), 1192-1231.
- Lynn, R., Allik, J., & Irving, P. (2004). Sex differences on three factors identified in Raven's SPM. *Intelligence*, 32, 411-424.
- McGregor, K., Kunda, M., & Goel, A. (2014). Fractal and Ravens. *Artificial Intelligence* 215, 1-23.
- O'Donoghue, D., Bohan, A., & Keane, M. (2006). Seeing Things: Inventive Reasoning with Geometric Analogies and Topographic Maps. *New Generation Computing* 24 (3), 267-288.
- Prade, H. & Richard, G. (2011). Analogy-Making for Solving IQ Tests: A Logical View. In *Procs. 19th International Conference on Case-Based Reasoning*, 561-566. London, UK: Springer.
- Ragni, M. & Neubert, S. (2014). Analyzing Raven's Intelligence Test: Cognitive Model, Demand, and Complexity. In H. Prade & G. Richard (Eds.) *Computational Approaches to Analogical Reasoning: Current Trends*, 351-370. Springer.
- Raven, J., Raven, J. C., & Court, J. (1998). *Manual for Raven's Progressive Matrices and Vocabulary Scales*. San Antonio, TX: Harcourt Assessment.
- Schwering, A., Krumnack, U., Kuhnberger, K-U, & Gust, H. (2009). Spatial cognition of geometric figures in the context of proportional analogies. In *Procs. Spatial Information Theory, Lecture Notes in Computer Science Volume 5756*, 18-35.
- Snow, R., Kyllonen, P., & Marshalek, B. (1984). The topography of ability and learning correlations. *Advances in the Psychology of Human Intelligence*, 2, 47-103.

Two Visual Strategies for Solving the Raven's Progressive Matrices Intelligence Test

Maithilee Kunda, Keith McGregor, and Ashok Goel

School of Interactive Computing, Georgia Institute of Technology
 {mkunda, keith.mcgregor}@gatech.edu, goel@cc.gatech.edu

Abstract

We present two visual algorithms, called the affine and fractal methods, which each solve a considerable portion of the Raven's Progressive Matrices (RPM) test. The RPM is considered to be one of the premier psychometric measures of general intelligence. Current computational accounts of the RPM assume that visual test inputs are translated into propositional representations before further reasoning takes place. We propose that visual strategies can also solve RPM problems, in line with behavioral evidence showing that humans do use visual strategies to some extent on the RPM. Our two visual methods currently solve RPM problems at the level of typical 9- to 10-year-olds.

Keywords: Analogy; intelligence tests; mental imagery; Raven's Progressive Matrices; visual reasoning.

Introduction

In previous work (Kunda, McGregor, & Goel, 2010), we presented two visual algorithms for solving problems from Raven's Progressive Matrices (RPM) intelligence tests, along with experimental results. Here, we summarize this recent work and discuss its implications for AI.

The RPM consists of geometric analogy problems, as shown in Figure 1; problems contain either 2x2 (like the example given in Figure 1) or more complex 3x3 matrices. Although the RPM is supposed to measure only eductive ability (Raven, Raven, & Court, 2003), its high correlation with other IQ tests has rendered it a premier psychometric measure of general intelligence, and it is widely used in clinical, educational, occupational, and scientific settings.

Computational accounts of problem solving on the RPM have generally assumed that visual inputs are translated into propositions before further reasoning takes place. Carpenter et al. (1990) implemented a production system that took hand-coded propositional descriptions of RPM inputs and then chose from predefined rules to solve each problem. Bringsjord and Schimanski (2003) used a theorem-prover to solve RPM problems stated in first-

order logic. Lovett, Forbus, & Usher (2007) combined automated sketch understanding with structure-mapping to solve RPM problems, which were represented using hand-drawn vector graphics. Cirillo and Strom (2010) also used vector graphics representations of RPM problems along with a set of pre-defined patterns to predict an answer. Finally, Rasmussen & Eliasmith (2011) used a spiking neuron model to induce rules for solving RPM problems represented as hand-coded propositional vectors.

Despite considerable differences in architecture and focus, all of these computational approaches have been similar in two respects. First, as mentioned earlier, they all rely on some translation process to convert visual test inputs into propositional representations, which then become the sole form of representation used by each system. Second, each system posits only one fundamental problem-solving strategy; individual differences in RPM performance are assumed (either implicitly or explicitly) to stem from quantitative variations in this strategy, rather than from qualitative differences in the strategy itself.

In contrast, Hunt (1974) proposed two qualitatively different strategies: Gestalt, using visual representations and perceptual operations, and Analytic, using propositions and logical operations. While neither RPM algorithm was

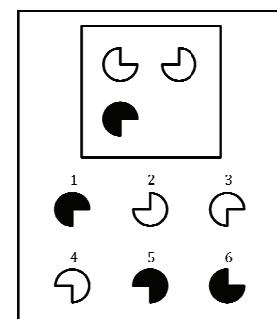


Figure 1: Example problem similar to one from the Standard Progressive Matrices (SPM) test. The correct answer is 5.

<p>For each base transform T:</p> <p>Apply T to Image A.</p> <p>Find best-match translation (t_x, t_y) between $T(A)$ and B, using Eq. (2).</p> <p>Find image composition operand X as follows:</p> <ul style="list-style-type: none"> Calculate similarity using Eq. (1) with: (1) $\alpha = 1, \beta = 1$ (2) $\alpha = 1, \beta = 0$ (3) $\alpha = 0, \beta = 1$ <p>Choose maximum similarity value.</p> <p>If max. is (1), then $X = 0$.</p> <p>If max. is (2), then $X = B - A$, and \oplus is image addition.</p> <p>If max. is (3), then $X = A - B$, and \oplus is image subtraction.</p> <p>The best-fit similitude transformation can then be specified as:</p> $[T_{\max} + (t_x, t_y)](A) \oplus X = B$	<p>Decompose D into a set of N smaller images $\{d_1, d_2, d_3, \dots, d_n\}$. These individual images are sets of points.</p> <p>For each image d_i:</p> <p>Examine the entire source image S for an equivalent image s_i such that a similitude transformation of s_i will result in d_i. This transformation will be a 3x3 matrix, as the points within s_i and d_i under consideration can be represented as the 3D vector $\langle x, y, c \rangle$ where c is the (grayscale) color of the 2D point $\langle x, y \rangle$.</p> <p>Collect all such transforms into a set of candidates C.</p> <p>Select from C the transform which most minimally achieves its work, according to some predetermined, consistent metric.</p> <p>Let T_i represent the chosen affine transformation of s_i into d_i.</p> <p>The set $T = \{T_1, T_2, T_3, \dots, T_n\}$ is the fractal encoding of the image D.</p>
---	--

Figure 2. Algorithm 1 (left): Affine method. Algorithm 2 (right): Fractal method.

implemented, a theoretical analysis suggested that both methods would be equally effective for certain problems.

In addition to Hunt's computational argument, there is evidence that humans use qualitatively different RPM strategies. Within-individual strategy differences have been studied as a function of problem type, primarily through factor analyses (Lynn et al., 2004), often dividing problems into those solvable with visual versus verbal operations.

Between-individual strategy differences have emerged in studies of autism. Whereas the RPM scores of typically developing (TD) individuals are highly correlated with full IQ scores, individuals with autism often show much higher RPM scores (Dawson et al., 2007), possibly because intact visual abilities in autism can be recruited to solve many RPM problems (Kunda & Goel, in press). Recent fMRI data of the RPM showed that individuals with autism had lower brain activation in areas associated with language and working memory and higher activation in visual areas than did TD individuals (Soulières et al., 2009).

Visual Methods for the Raven's Test

We have developed two different RPM algorithms that we call the "affine" method and the "fractal" method. Unlike previous computational models of the RPM, these methods use image transformations to solve RPM problems directly, without first converting visual inputs into propositions. We define a representation as being purely visual if it captures image information only at the pixel level, i.e. as a spatial array of individual color/intensity values. A representation is propositional if it encodes higher-level visual entities as propositions, e.g. as labeled lines, shapes, textures, etc.

Figure 2 outlines the core mechanisms for both visual algorithms. Due to space constraints, we present only brief descriptions of these algorithms below; more details can be found in (Kunda, McGregor, & Goel, 2010).

At the core of these methods are affine transformations, and in particular similitude transforms, which can be represented as compositions of dilation, orthonormal transformation, and translation. We presently use the

identity transform, horizontal and vertical reflections, and 90° , 180° , and 270° rotations, composed with translation. There is evidence that human visual processing can apply some of these types of transformations to mental images, or at least operations that are computationally isomorphic (Kosslyn, Thompson, & Ganis, 2006).

Similarity also lies at the core of both methods, as calculated using the ratio model (Tversky, 1977):

$$\text{similarity}(A, B) = \frac{f(A \cap B)}{f(A \cap B) + \alpha f(A - B) + \beta f(B - A)}$$

In Eq. (1), f represents some function over features in each of the specified sets, and α and β are weights for the non-intersecting portions of the sets A and B. If α and β are both set to one, then this equation becomes:

$$\text{similarity}(A, B) = \frac{f(A \cap B)}{f(A \cup B)}$$

Eq. (2) is used in both methods, and it yields maximal similarity when A is equal to B. In contrast, if α is one and β is zero, it yields maximal similarity when A is a proper subset of B. If α is zero and β is one, then maximal similarity is found when B is a proper subset of A.

The Affine Method

The affine method assumes that 1) elements within a row or column in an RPM problem matrix are related by similitude transforms, and 2) analogical relationships exist, in the form of identical similitude transforms, across parallel rows or columns of the matrix. Each similitude transform is represented as a combination of three image operations: base transform, translation, and composition.

First, the algorithm determines which transform best fits any complete row or column in the matrix; Algorithm 1 in Figure 2 shows how, for images A and B, the "best-fit" transform is found. Then, this transform is applied to whichever parallel row/column contains the missing entry to generate a guess image for the answer. Finally, this guess is compared to each answer choice, using Eq. (2), and the best match is chosen as the answer.

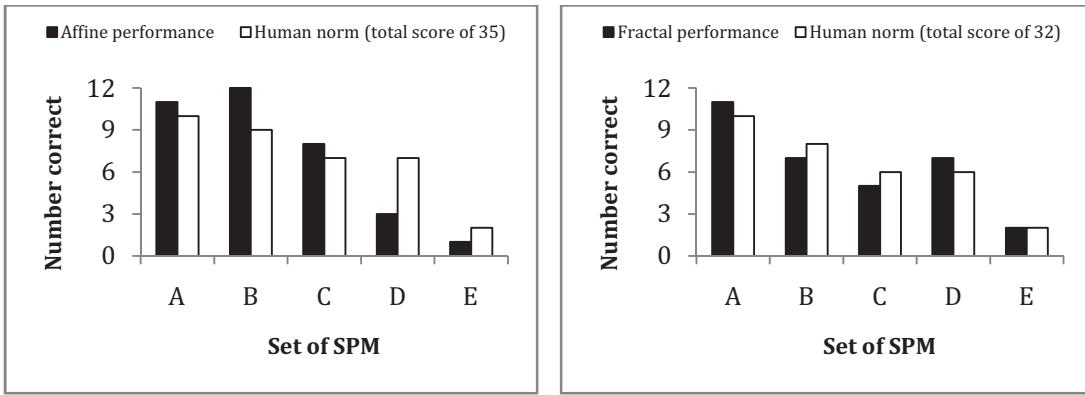


Figure 3: Breakdown of affine (left) and fractal (right) results across sets in the SPM. Also shown is the expected score breakdown for total scores of 35 and 32, from normative human data (Raven, Raven, & Court, 2003).

The Fractal Method

Like the affine method, the fractal method seeks to find a re-representation of the images within an RPM problem as a set of similitude transforms. Unlike the affine method, the fractal method seeks representations at a significantly finer partitioning of the images, and uses features derived from these resulting “fractal” representations to determine similarity for each possible answer, simultaneously, across the bulk of relationships present in the problem.

The mathematical derivation for the process of fractal image representation expressly depends on the notion of real world images, i.e. images that are two dimensional and continuous (Barnsley & Hurd, 1992), and draws upon the 1) repetition and 2) similarity at different scales that are found in such images. Fractal representations seek to describe images in terms other than those of shapes or traditional graphical elements—i.e. terms that capture this observed similarity and repetition alone. Computationally, determining fractal representations uses the fractal encoding algorithm, shown in Algorithm 2 in Figure 2.

Once fractal representations have been calculated for each pair of images in the problem matrix, Eq. (2) is used to calculate similarity between all pairwise relationships in the matrix and those calculated with the given answer choices, using features derived from the fractal encodings. Whichever answer choice yields the most similar fractal representations across all pairwise relationships is chosen as the final answer (McGreggor, Kunda, & Goel, 2010).

Results

We tested both the affine and fractal methods on the complete Standard Progressive Matrices (SPM) test. The SPM consists of 60 problems divided into five sets labeled A-E. To obtain visual inputs for the algorithms, we first scanned a paper copy of the SPM, aligned each page squarely, and then divided each problem into separate

image files for each matrix entry and answer. No further image processing was performed on these images.

The affine algorithm correctly solved 35 of 60 problems on the SPM, as shown in Figure 3. For children in the U.S., this total score corresponds to the 50th percentile for 10½-year-olds (Raven, Raven, & Court, 2003). The fractal algorithm correctly solved 32 of 60 problems, which corresponds to the 50th percentile for 9½-year-olds.

We also looked at the performance of both methods as a function of problem type on the SPM; we used the results from a factor analysis to divide problems into those loading on “gestalt continuation,” “visuospatial,” or “verbal-analytic” factors (Lynn, Allik, & Irving, 2004). Figure 4 shows the performance of both methods with respect to problem type. Both the affine and fractal methods perform most strongly on gestalt problems, slightly less so visuospatial problems, and significantly less so on problems requiring verbal-analytic reasoning, though problem difficulty represents a potential confound.

Discussion

We presented results from (Kunda, McGregor, & Goel, 2010) describing two algorithms that use purely visual representations to solve more than half of the problems on the Raven’s SPM test. These results are significant not only because we have shown purely visual methods to be surprisingly powerful in addressing a standard test of general intelligence, which is itself an accomplishment according to the psychometric AI school of thought (Brnsjord & Schimanski, 2003), but also because they illustrate the capabilities of visual methods in general.

As previous RPM models have shown, as well as other visual analogy work in AI (Davies, Yaner, & Goel, 2008), visuospatial knowledge alone, represented propositionally, can be sufficient to solve certain analogy problems. In this work, we have shown that it is not necessary to translate

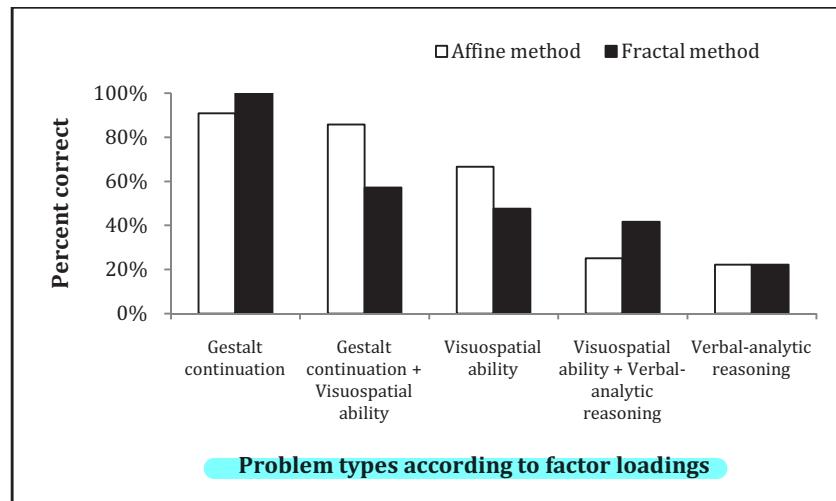


Figure 4: Breakdown of affine and fractal algorithm results on the SPM by problem type. Problem breakdowns were obtained from a factor-analytic study of human performance (Lynn, Allik, & Irving, 2004).

images into propositions at all for certain reasoning to take place. The analogical (i.e. having a structure corresponding to what is represented) properties of real-world images, including their amenability to affine transforms, repetition, and self-similarity, appear sufficient to support higher-level reasoning, and the ratio model of similarity is one instantiation of a comparison mechanism that does not rely on the extraction of propositional features.

In general, the results of this work support the view of perception as a first-class object for reasoning in AI systems. Many open questions remain for the study of visual strategies on the RPM, such as what role other visual operations like convolution or deformation might play, and how visual and propositional strategies might be seamlessly bridged in a single cognitive agent.

Acknowledgments

This research has been supported by an NSF grant (IIS Award #0534266), “Multimodal Case-Based Reasoning in Modeling and Design,” by ONR through an NDSEG fellowship, and by the NSF GRFP fellowship program.

References

- Barnsley, M. F., & Hurd, L. P. (1992). Fractal Image Compression. Boston, MA: A.K. Peters.
- Bringsjord, S., & Schimanski, B. (2003). What is artificial intelligence? Psychometric AI as an answer. *IJCAI*, 18, 887–893.
- Carpenter, P., Just, M., & Shell, P. (1990). What one intelligence test measures: A theoretical account of the processing in the Raven Progressive Matrices test. *Psychological Rev.*, 97, 404-31.
- Cirillo, S., & Strom, V. (2010). An anthropomorphic solver for Raven’s Progressive Matrices (No. 2010:096). Goteborg, Sweden: Chalmers University of Technology.

Davies, J., Goel, A., & Yaner, P. (2008). Proteus: Visuospatial analogy in problem solving. *Knowledge-Based Systems*, 21, 636-654.

Dawson, M., Soulières, I., Gernsbacher, M., & Mottron, L. (2007). The level and nature of autistic intelligence. *Psychological Science*, 18, 657-662.

Hunt, E. (1974). Quote the raven? Nevermore! In L. Gregg (Ed.), *Knowledge and Cognition* (pp. 129–158). Hillsdale, NJ: Erlbaum.

Kunda, M., & Goel, A. (in press). Thinking in Pictures as a cognitive account of autism. *J. Autism and Dev. Disorders*.

Kunda, M., McGregor, K., & Goel, A. (2010). Taking a look (literally!) at the Raven’s intelligence test: Two visual solution strategies. In *Proc. 32nd Conf. Cog. Sci. Society* (pp. 1691-1696)

Lovett, A., Forbus, K., & Usher, J. (2007). Analogy with qualitative spatial representations can simulate solving Raven’s Progressive Matrices. In *Proc. 29th Conf. Cog. Sci. Society* (pp. 449-454).

Lynn, R., Allik, J., & Irving, P. (2004). Sex differences on three factors identified in Raven’s SPM. *Intelligence*, 32, 411-424.

McGregor, K., Kunda, M., & Goel, A. (2010). A fractal approach towards visual analogy. In *Proc. 1st International Conf. Computational Creativity*.

Rasmussen, D., & Eliasmith, C. (2011). A neural model of rule generation in inductive reasoning. *Topics in Cognitive Science*, 3, 140-153.

Raven, J., Raven, J. C., & Court, J. (2003). *Manual for Raven’s Progressive Matrices and Vocabulary Scales*. San Antonio, TX: Harcourt Assessment.

Soulières, I., Dawson, M., Samson, F., Barbeau, E., Sahyoun, C., Strangman, G., Zeffiro, T., & Mottron, L. (2009). Enhanced visual processing contributes to matrix reasoning in autism. *Human Brain Mapping*, 30, 4082-107.

Tversky, A. (1977). Features of similarity. *Psychological Review*, 84, 327-352.

KBAI Ebook: Knowledge-based Artificial Intelligence



KBAI: CS7637 course at Georgia Tech:

Course Creators and Instructors: Ashok Goel, David Joyner.

[Click here for Course Details](#)

Electronic Book (eBook) Designers: Bhavin Thaker, David Joyner, Ashok Goel.

Last updated: October 6, 2016



Ashok Goel



David Joyner



Bhavin Thaker

Copyright: Ashok Goel and David Joyner, Georgia Institute of Technology.

All rights reserved. No part of this document may be reproduced, stored in any retrieval system, or transmitted in any form or by any means without prior written permission.

YouTube Playlists:

[Part 1 of 5](#) | [Part 2 of 5](#) | [Part 3 of 5](#) | [Part 4 of 5](#) | [Part 5 of 5](#)

NOTE: Lessons 02, 04, 14 and 25 have a correspondence problem between YouTube links, transcripts, and slides. Additionally, the following videos have known incomplete transcripts:

Lesson 3 - Exercise: Constructing Semantic Nets I,

Lesson 5 - Exercise: Block Problem I,

Lesson 24 - Example: Goal-Based Autonomy, and

Lesson 25 - Raven's Progressive Matrices.

These will get fixed when the YouTube playlist is fixed by Udacity and reloaded into this KBAI Ebook.

Lesson 01 - Introduction to Knowledge-Based AI



To understand the intelligence functions at a fundamental level, I believe, would be a scientific achievement on the scale of nuclear physics, relativity, and molecular genetics.
– James Albus.

Education is not the piling on of learning, information, data, facts, skills, or abilities – that's training or instruction – but is rather making visible what is hidden as a seed.
– Thomas More.

01 - Introductions

[Click here to watch the video](#)

Lesson Preview

- What is knowledge-based artificial intelligence?
- How does it fit into the rest of artificial intelligence?
- What can I expect to learn from this course?
- What is the structure of this course?

is for education and especially for using modern technology to deliver individualized personal educational experiences. It would be very difficult in very large classrooms. As we'll see AI is not just the subject of this course, but it's all a tool we're using to teach this course. We had a lot of fun putting this course together. We hope you enjoy it as well. We think of this course as an experiment as well. We want to understand how students learn in online classrooms. So if you have any feedback please share it with us.

02 - Preview

[Click here to watch the video](#)

Figure 1: Introductions

Hello, and welcome to CS 7637, knowledge-based artificial intelligence/cognitive systems. My name is Ashok Goel. My name is David Joyner. I'm a Professor of Computer Science and Cognitive Science at Georgia Tech. I've been teaching knowledge-based AI for about 25 years. I've been doing research in this area for about 30. My personal passion is for computational creativity. Breathing air agents that human like and creative in their own right. I'm, of course, developer with Udacity and I'm also finishing up my own PhD dissertation here at Georgia Tech with Ashok as my advisor. My personal passion

What are some things Watson must be able to do to participate in Jeopardy?

- Read the clue
- Search its knowledge base
- Decide on an answer
- Properly phrase that answer

Figure 2: Preview

So welcome to 7637 Knowledge Based AI. At the beginning of each lesson, we'll briefly intro-

duce a topic as shown in the graphics to the right. We'll also talk about how the topic fits into the overall curriculum for the course. Today, we'll be discussing AI in general, including some of the fundamental conundrums and characteristics of AI. We will describe four schools of AI, and discuss how knowledge based AI fits into the rest of AI. Next, we'll visit the subtitle of the course, Cognitive Systems, and define an architecture for them. Finally, we'll look at the topics that we'll cover in this course in detail.

03 - Conundrums in AI

[Click here to watch the video](#)

- Fundamental Conundrums of Artificial Intelligence**
- Intelligent agents have limited resources.
 - Computation is local, but problems have global constraints.
 - Logic is deductive, but many problems are not.
 - The world is dynamic, but knowledge is limited.
 - Problem solving, reasoning, and learning are complex, but explanation and justification are even more complex.

it, we'll discuss it later in the later in the class. Conundrum number four. The world is dynamic, knowledge is limited, but an AI agent must always begin with what it already knows. How then can an AI agent ever address a new problem? Conundrum number five. Problem solving, reasoning, and learning are complex enough, but explanation and justification add to the complexity. How then can we get an AI agent to ever explain or justify its decisions?

04 - Characteristics of AI Problems

[Click here to watch the video](#)

Characteristics of AI Problems

- Knowledge often arrives incrementally.
- Problems exhibit recurring patterns.
- Problems have multiple levels of granularity.
- Many problems are computationally intractable.
- The world is dynamic, but knowledge of the world is static.
- The world is open-ended, but knowledge is limited.

Figure 3: Conundrums in AI

Let's start a recognition today. We're discussing some of the biggest problems in AI. We obviously are not going to solve all of them today, but it's good to start with a big picture. AI has several conundrums, I'm going to describe five of the main ones today. Conundrum number one. All intelligent agents have little computational resources, processing speed, memory size, and so on. But most interesting AI problems are computationally intractable. How then can we get AI agents to give us near real time performance on many interesting problems? Conundrum number two. All competition is local, but most AI problems have global constraints. How then can we get AI agents to address global problems using only local computation? Conundrum number three. Computation logic is fundamentally deductive, but many AI problems are abductive or inductive in their nature. How can we get AI agents to address abductive or inductive problems? If you do not understand some of these terms, like abduction, don't worry about

Figure 4: Characteristics of AI Problems

I hope our discussion of the big problems in AI didn't scare you off, let's bring the discussion down, closer to work. And talk about a few fundamental characteristics of AI problems. Number one, in many AI problems, data arrives incrementally not all the data comes right at the beginning. Number two, problems often have a recurring pattern, the same kinds of problems occur again and again. Number three, problems occur at many different levels of abstraction. Problem number four, many interesting AI problems are computationally intractable. Number five, the world is dynamic, it's constantly changing but knowledge of the world is relative to static. Number six, the world is open ended but knowledge of the world is relatively limited. So, the question then becomes, how can we design air agents that can address air problems with these characteristics, those are the challenges we'll discuss in this course

05 - Characteristics of AI Agents

[Click here to watch the video](#)

- Characteristics of AI Agents*
- Agents have limited computing power.
 - Agents have limited sensors.
 - Agents have limited attention.
 - Computational logic is fundamentally deductive.
 - AI agents' knowledge is incomplete relative to the world.

Figure 5: Characteristics of AI Agents

In addition to AI problems having several characteristics, AI agents too have several properties. Property number one. AI agents, have only a limited computing power, processing speed, memory size, and so on. Property number two. AI agents have limited sensors, they cannot perceive everything in the world. Property number three. AI agents have limited attention, they cannot focus on everything at the same time. Property number four. Computational logic is fundamentally deductive. Property number five. The world is large, but AI agents' knowledge of the world is incomplete relative to the world. So, the question then becomes, how can AI agents with such bounded rationality address open-ended problems in the world?

06 - Exercise What are AI Problems

[Click here to watch the video](#)

Which of these are AI problems?

- Answering questions on Jeopardy.
- Configuring the dimensions for the basement of a new house.
- Tying shoelaces.
- Deciding on a route to a new destination.
- Making sense of a news broadcast.
- Designing a robot that walks on water.
- Establishing whether a flower pot can be used as a drinking cup.
- Deciding whether or not a new animal is a bird.

Figure 6: Exercise What are AI Problems

Now that we have talked about the characteristics of AI, agents in AI problems. Let us talk a little about for what kind of problems might you build in AI agent. On the right are several tasks. Which are these AI problems? Or to put

it differently, for which of these problems would you build an AI agent to solve?

07 - Exercise What are AI Problems

[Click here to watch the video](#)

David, which one of these do you think are AI problems? Science has said that all of these are AI problems. All of these are things that we humans do on a fairly regular basis. And if the goal of artificial intelligence is to recreate human intelligence, then it seems like we need to be able to design agents that can do any of these things. I agree. In fact, during this class we'll design AI agents that can address each of these problems. For now, let us just focus on the first one. How to design an AI agent that can answer Jeopardy questions.

08 - Exercise AI in Practice Watson

[Click here to watch the video](#)

Let's start with looking at an example of an AI agent in action. Many of you are familiar with Watson, the IBM program that plays Jeopardy. Some of you may not be, and that's fine, we'll show an example in a minute. When you watch Watson in action, try to think. What are some of the things Watson must know about? What are some of the things that Watson must be able to reason about in order to play Jeopardy? Write them down. And anytime you feel the pain, hey, this guy refrains. Don't carry the world upon your shoulders. Watson? Who is, Jude? Yes. Olympic Oddities, for 200. Milorad Cavic almost upset this man's perfect 2008 Olympics, losing to him by one-hundredth of a second. Watson. Who is Michael Phelps. Yes, go. Name the decade for 200. Disneyland opens and the peace symbol is created. Ken. What are the 50s? Yes. Final Frontiers for 1,000, Alex. Tickets aren't needed for this event, a black hole's boundary from which matter can not escape. Watson. What is event horizon?

09 - Exercise AI in Practice Watson

[Click here to watch the video](#)

David, what did you write it down? So I said that the four fundamental things a Watson must be able to do to play Jeopardy are first

read the clue, then search through it's knowledge base, then actually decide on it's answer, and then phrase it's answer in the form of a question. That's right. And during this course, we'll discuss each part of David's answer.

10 - What is Knowledge-Based AI

[Click here to watch the video](#)

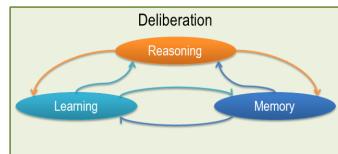


Figure 7: What is Knowledge-Based AI

Let us look at the processes that Watson may be using a little bit more closely. Clearly Watson is doing a large number of things. It is trying to understand natural language sentences. It is trying to generate some natural language sentences. It is making some decisions. I'll group all of these things broadly under reasoning. Reasoning is a fundamental process of knowledge based data. A second fundamental process of knowledge based AIs learning. What simply is learning also? It perhaps gets a right answer to some questions, and stores that answer somewhere. If it gets a wrong answer, and then once it learns about the right answer, it stores the right answer also somewhere. Learning to is a fundamental process of knowledge based AI. A third fundamental process of knowledge based ai is memory. If you're going to learn something, that knowledge that you're learning has to be stored somewhere, in memory. If you're going to reason using knowledge, then that knowledge has to be accessed from somewhere, from memory. From memory process it will store, what we learn as well as provide access to knowledge it will need for reasoning. These three forms of processes of learning, memory, and reasoning are intimately connected. We learn, so that we can reason. The result of reasoning often. Result in additional

learning. Once we learn, we can store it in memory. However, we need knowledge to learn. The more we know, the more we can learn. Reasoning requires knowledge that memory can provide access to. The results of reasoning can also go into memory. So, here are three processes that are closely related. A key aspect of this course on knowledge based AI is that we will be talking about theories of knowledge based AI that unify reasoning, learning, and memory. And sort of, discussing any one of the three separately as sometimes happens in some schools of AI. We're going to try to build, unify the concept. These 3 processes put together, I will call them deliberation. This deliberation process is 1 part of the overall architecture of a knowledge based AI agent. This figure illustrates the older architecture of an AI agent. Here we have input in the form of perceptions of the world. And output in the form of actions in the world. The agent may have large number of processes that map these perceptions to actions. We are going to focus right now on deliberation, but the agent architecture also includes metacognition and reaction, that we'll discuss later

11 - Foundations The Four Schools of AI

[Click here to watch the video](#)

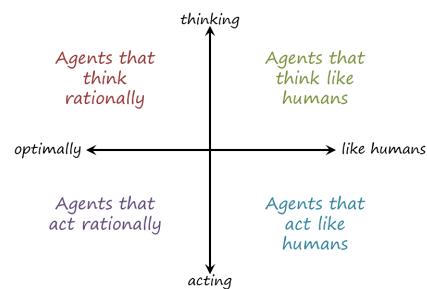


Figure 8: Foundations The Four Schools of AI

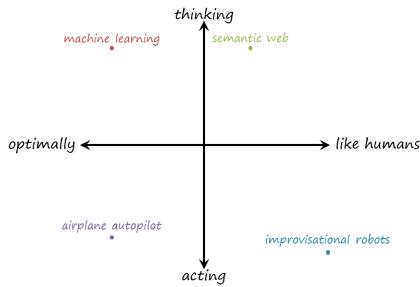


Figure 9: Foundations The Four Schools of AI

Another way of understanding what is knowledge based AI, is to contrast it with the other schools of thought in AI. We can think in terms of a spectrum. On one end of the spectrum, is acting. The other end of the spectrum is thinking. As an example, when you're driving a car, you're acting on the world. But when you are planning what route to take, you're thinking about the world. There is a second dimension for distinguishing between different schools of thought of AI. At one end of the spectrum we can think of AI agents that are optimal. At the other end of the spectrum, we can think of air agents that act and think like humans. Humans are multifunctional, they have very robust intelligence. That intelligence need not be optimal relative to any one task, but it's very general purpose, it works for a very large number of tasks. Were as we can pick up here, agents on the other side which are optimal for a given task. Given these 2 axis we get 4 quadrants. Starting from the top left and going counter clockwise, here are Agents that think optimally, Agents that act optimally, Agents that act like humans. And agents that think like humans. In this particular course in knowledge based AI, we're interested in agents that think like humans. Let us take a few examples to make sure that we understand this four quadrants world. Here are some well known computational techniques. Consider many machine learning algorithms. These algorithms analyse large amounts of data, and determine patterns of the regularity of that data. Well I might think of them as being in the top left quadrant. This is really doing thinking, and they often are optimal, but they're not necessarily human like. Airplane autopilots. They

would go under acting optimally. They're suddenly acting in the world, and you want them to act optimally. Improvisational robots that can perhaps dance to the music that you play, they're acting, and they are behaving like humans, dancing to some music. Semantic web, a new generation of web technologies in which the web understands the various pages, and information on it. I might put that under thinking like humans. They are thinking. Not acting in the world. And is much more like humans, than, let's say, some of the other computational techniques here. If you're interested in reading some more about these projects, you can check out the course materials. Where we've provided some recent papers on these different computational techniques. There's a lot of cutting edge research going on here at Georgia Tech and elsewhere, on these different technologies. And if, if you really are interested in this, this is something where we're always looking for contributors.

12 - Exercise What is KBAI[Click here to watch the video](#)

So one thing that many students in this class are probably familiar with is Sebastian Thrun's Robotics class on autonomous vehicles. David, where do you think an autonomous vehicle would fall on the spectrum? So it seems to me like an autonomous vehicle definitely moves around in the world so it certainly acts in the world. And driving is a very human-like behavior so I'd say that it acts like a human. What do you think? Do you agree with David?

13 - Exercise What is KBAI[Click here to watch the video](#)*Is David right?*

- Yes, because an autonomous car mimics a human behavior.
- No, because an autonomous car just does the thinking; something else does the acting.
- No, because it's more important for an autonomous vehicle to act correctly than to act like humans.
- No, because an autonomous car just thinks instead of acts, and the thinking isn't even human-like.

Figure 10: Exercise What is KBAI

David, do we really care whether or not the autonomous vehicle thinks and acts the way we do? I guess, now that you mention it, as long as the vehicle gets me to my destination and doesn't run over anything on the way, I really don't care if it does think the way I do. And if you look at the way I drive, I really hope it doesn't act the way I do. So the autonomous vehicle may really belong to the acting rationally side of the spectrum. At the same time, looking at the way humans write might help us design a robot. And looking at the robot design might help us reflect on human cognition. This is one of the patterns of knowledge-based data.

14 - Exercise The Four Schools of AI

[Click here to watch the video](#)

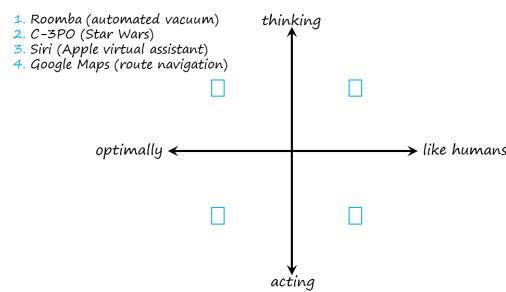


Figure 11: Exercise The Four Schools of AI

Let us do an exercise together. Once again, we have the four quadrants shown here, and at the top left are four compression artifacts. I'm sure you're familiar with all four of them. C-3PO is a fictitious artifact from Star Wars. Can we put these four artifacts in the quadrants to which they best belong?

15 - Exercise The Four Schools of AI

[Click here to watch the video](#)

What do you think about this David? So starting with Roomba, I would put Roomba in the bottom left. It definitely acts in the world. But it definitely doesn't act like I do. It criss-crosses across the floor until it vacuums everything up. So we're going to say that's acting optimally. C-3PO is fluent in over 6 million forms of communication. And that means that it interacts with human and other species very often.

In order to do that, it has to understand natural sentences and put its own knowledge back into natural sentences. So, it has to act like humans. Apple's virtual assistant, Siri, doesn't act in the world. So she is more on the thinking end of the spectrum but, like C-3PO, she has to interact with humans. She has to read human sentences and she has to put her own responses back into normal vernacular. So we're going to say that she thinks like humans. Google Maps plots your route from your origin to your destination. So it's definitely doing thinking, it's not doing any acting in the world. But we don't really care if it does the route planning like we would do it. So we would say it does its route planning optimally. It takes into consideration traffic. Current construction, different things like that, where we would probably think of the routes we have taken in the past. So Google Maps thinks optimally. That is a good answer David, I agree with you, but not here that some aspects of Siri, may well belong in some of the other quadrants. So, putting under 3 sounds plausible. But Siri might also be viewed as, perhaps, acting when it gives you a response. Siri, some aspects of Siri might also be optimal, not necessarily like humans. So if you'd like to discuss where these technologies belong on these spectrums or, perhaps, discuss where some other AI technologies that you're familiar with belong on these spectrums, feel free to head on over to our forums where you can bring up your own technologies and discuss the different ways in which they fit into the broader school of AI.

16 - What are Cognitive Systems

[Click here to watch the video](#)

I'm sure you have noticed that this class has a subtitle, cognitive systems. Let's talk about this term and break it down into its components. Cognitive, in this context, means dealing with human-like intelligence. The ultimate goal is to dwell up human level, human-like intelligence. Systems, in this context, means having multiple interacting components, such as learning, reasoning and memory. Cognitive systems, they are systems that exhibit human level, human-

like intelligence through interaction among components like learning, reasoning and memory. Thus, on a spectrum, what we'll discuss in this class will definitely lie on the right side of the spectrum, on the human side. We will be talking about thinking and acting, but we will always be concerned with human cognition.

17 - Cognitive System Architecture

[Click here to watch the video](#)

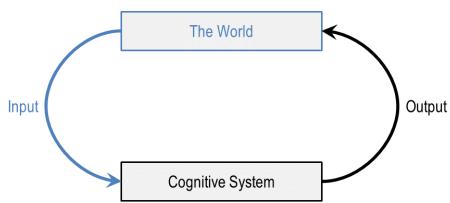


Figure 12: Cognitive System Architecture

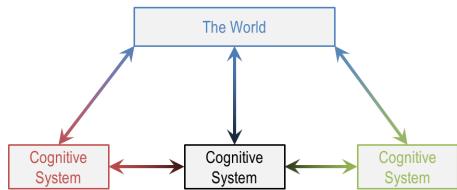


Figure 13: Cognitive System Architecture

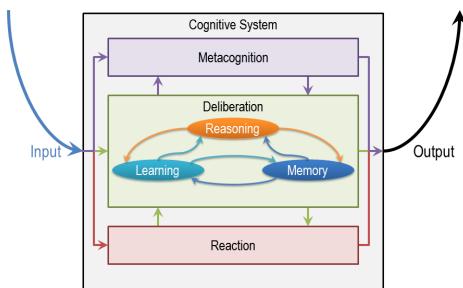


Figure 14: Cognitive System Architecture

So let us take a look at what is a cognitive system. Notice that I'm using the term cognitive system and not the term knowledge-based AI

agent. I could have used that term also. When we talk about knowledge-based AI agent, then we could take two views. One view is that we are going to build a knowledge-based AI system, which need not be human like. Another view is that the knowledge based AI agent that we will build will be human-like. The cognitive system is situated in the world. Here by the world I mean the physical world. For example, the world that I am interacting with right now, with this screen in front of me and this microphone. This world is perceptionate. There's an example, the percept something being a straight line or a color of some object. Or the smoothness of some of the texture of some object. This perceptionate around the world and cognitive system is using sensors to perceive this percept. That's the input of the cognitive system. The cognitive system also has some actuators. So, for example, I have fingers that I'm using right now to point to things. And a cognitive system uses actuators to carry out actions on the world. Cognitive system then is taking perceptor's input and giving actions as output. So far, we've talked about a single cognitive system. But of course one can have multiple cognitive systems. These multiple cognitive systems can interact with each other. Just like a cognitive system situated in a physical world, it is also situated in a social world. Let us now zoom into the inside of a cognitive system. What is the architecture of a cognitive system? So the cognitive system takes as input certain percepts about the world. It has a task of giving as output actions of the world. The question then becomes, how can these percepts be mapped into actions? One way of mapping them is that we will do a direct mapping. These percepts will be directly mapped into actions. Let's take an example. Imagine that you're driving a car, and the brake lights of the car in front of you, become bright red. Should that happen, you will then press on the brakes of your car. Well, that is an example of a reactive system. The percepts were that the break lights on the car in front of you became bright red and the action was that you pressed on your own brakes. In doing so, you may not have planned. This is now a direct mapping of percept into actions. Alternatively, con-

sider a slightly different problem. Again you're driving your car on the highway, but as you're trying to drive on the highway your task this time is to change lanes. Now, in order to change lanes, again you may look around and look at the percept of the road. There are other cars on the road, for example, and you need to take some action that will help you change lanes. This time you may actually deliberate, you may actually look at the goal that you have as well as the percepts of the environment and come up with a plan that will tell you what action to take. As we discussed in the last lesson, the deliberation itself has a number of components in it. Three of the major components that we'll be studying in this class are learning, reasoning, and memory. These three components interact with each other in many interesting ways that we will decipher as we go along. Now, deliberation was reasoning about the world around us. So if I take that example again of changing lanes, as I'm driving on the highway, then I'm reasoning about the world around me. Where are the other cars? Should I change lanes to the left or to the right. Metacognition on the other hand, the third layer here, has to do with reasoning about the internal mental world. So metacognition reasons about the deliberation. Or metacognition can also reason about reaction. Let us take an example of the metacognition also. Imagine again that I had to change lanes. And I did, as I changed lanes to the left, the cars behind me honk because I did not leave enough space for the car that was already moving on the left lane. In that case I know that the lane changing did not go very smoothly. I may now think about my own actions in the world, about the deliberation that led to those actions, and I may then decide to change or reconfigure, or repair the deliberation that led to that sub-optimal plan for changing the lanes. That is an example of metacognition. So now I have this three layered architecture, reaction, deliberation, metacognition. Note that we have defined intelligence in a way, intelligence here is about mapping percepts in the world, interactions in the world. Intelligence is about selecting the right kind of action given a particular state of the world. But there are many different

ways in which we can map the percepts into actions. Purely reactive, deliberative, or also entailing metacognition on the deliberation and the reaction. This then is the overall architecture of the cognitive system. This is called a three layered architecture. We'll be returning to this architecture many times in this course.

18 - Topics in KBAI

[Click here to watch the video](#)

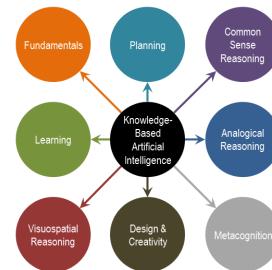


Figure 15: Topics in KBAI

Fundamentals

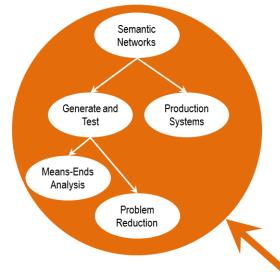


Figure 16: Topics in KBAI

Planning

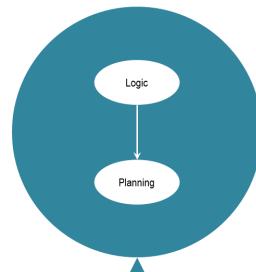


Figure 17: Topics in KBAI

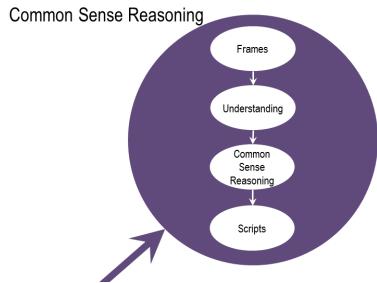


Figure 18: Topics in KBAI

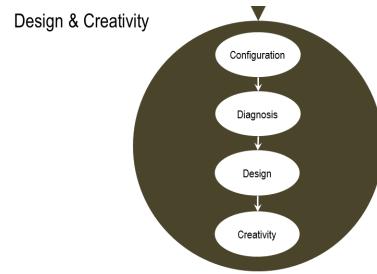


Figure 22: Topics in KBAI

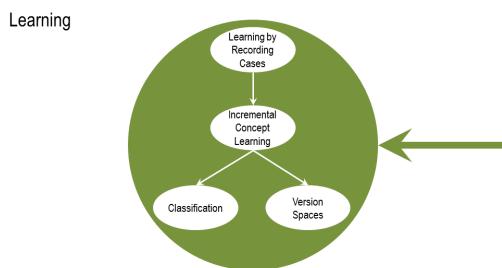


Figure 19: Topics in KBAI

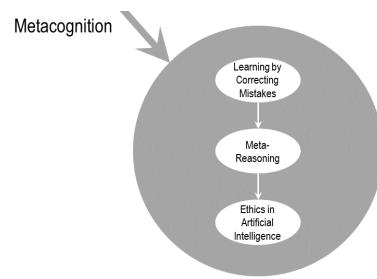


Figure 23: Topics in KBAI

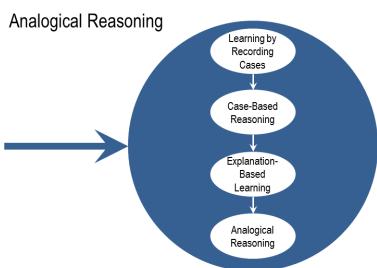


Figure 20: Topics in KBAI

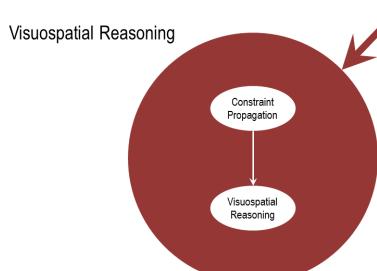


Figure 21: Topics in KBAI

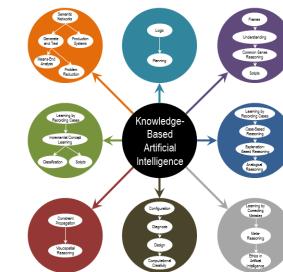


Figure 24: Topics in KBAI

We have organized the materials in this course, into eight major units, this chart illustrates those eight units. So starting from the top left, the first unit has to do with Fundamentals of presentation and recent, Panning, Common Sense Reasoning. Analogical Reasoning, Metacognition that we just talked a little about, Design & Creativity, Visuospatial Reasoning, and Learning. Now let's look at each of these circles, one at a time. So in the first part, dealing with the Fundamentals of this course. We'll be dealing with certain, knowledge representations, and reasoning strategies. Two of the major knowledge representations that we'll discuss in the first part of this course, are called

Semantic Networks, and Production Systems. Three of the reasoning strategies, are called Generate and Test, Means-End Analysis, and Problem Reduction. Note that, the arrows here imply an ordering. There is an ordering. In that, when we are discussing our Production Systems, we might allude to things that we are discussing in the Semantic Networks. Similarly, when we discuss Means-End Analysis, we might allude to things that we are discussing in Generate and Test. However. It is important to note also, that these three methods are completely independent from each other. It's just that we are going to discuss them, in the order shown here. Similarly these two knowledge representations, are independent from each other. It's simply that in this course, we'll discuss them in this order. So the second major unit in this course. Pertains to Planning. Planning is kind of problem solving activity whose goal is to come up with plans, for achieving one or more goals. Before we discuss Planning, we'll discuss Logic as a knowledge representation. This knowledge representation, will then enable us to discuss Planning in a systematic way. The third major unit in this course is common sense reasoning. Common Sense Reasoning, pertains to reasoning about every day situations in the world. As an example, I may give you the input, John gave the book to Mary. Note the input, does not specify who has the book at the end. But, you can draw that inference easily. That is an example of Common Sense Reasoning. In our course, we'll discuss both knowledge representations like frames, as well as methods for doing Common Sense Reasoning. As we discussed earlier, when we were talking about the architecture of a cognitive system, Learning is a fundamental process within deliberation. And therefore, we will be visiting the issue of Learning many, many times throughout this course. However, we also have a unit on Learning, which has several topics in it. There are other topics in Learning, that do not show up in this particular. Circle here but are distributed throughout the course. Another major unit in our course is Analogical Reasoning. Analogical Reasoning is, reasoning about novel problems or novel situations, but, analogic to

what we know about familiar problems, or familiar situations. As I mentioned earlier, Learning is distributed throughout this course. Therefore Learning comes here in Analogical Reasoning also. In fact Learning by recording cases appeared in the Learning topic as well as here, and you can see explanation based Learning occurring here. Visuospatial Reasoning is another major unit in our course. Visuospatial Reasoning pertains to reasoning with visual knowledge. As an example, I might draw a diagram and reason with the diagram. That's an example of Visualspatial Reasoning. In the context of Visualspatial Reasoning, we're talking both about Constraint Propagation, and using that to do Visualspatial Reasoning. Design & Creativity is the next topic in our course. We want to build AI systems, that can deal with novel situations, and come up with creative solutions. Design is an example of a complex task which can be very, very creative, where we are discussing a range of topics in the context of Design & Creativity. So in the next topic in our courses Metacognition, we have already come across a notion in Metacognition, when we were talking about the architecture of the cognize system. Metacognition pertains to thinking about thinking. And we'll discuss a range of topics and then, we will end the course by talking about Ethics in Artificial Intelligence. This figure illustrates all the eight major units once again, as well as the topics within each major unit. I hope this will give you mental map of the organization of the course as a whole. In preparing this course, we came up with a ordering of the topics, which will interleave many of these topics. So we will not do the entire first unit, before we go to the entire second unit and so on. Instead, we will do some parts of first unit, then go to some other part that follows conceptually from it, and so there will be some interleaving among these topics. And one aspect of the personalization is, that you are welcome to go through these topics in your own chosen order. You don't have to stay, with the kind of order that we'll be using. This is an exciting agenda. I hope you are as excited as I am. There are very few, opportunities where we can talk about exotic topics, like Analogical Reason-

ing, and Creativity, and Metacognition. And, in this particular course, we'll talk about all of them together.

19 - Wrap Up

[Click here to watch the video](#)

To recap...

- Conundrums and characteristics
- Four schools of AI
- What is KBAI?
- Cognitive Systems
- Topics in AI

Figure 25: Wrap Up

So at the end of every lesson, I will briefly recap what we talked about during that lesson and try to tie it into some future topics. Today, we started off by talking about the central conundrums and characteristics of AI. This may have connected with some of your previous experience with other AI classes, like machine learning in AI for robotics. We then talked about the four schools of AI, and we talked about knowledge-based AI more specifically, what is it and where does it fit in with the other schools? Then we talked about cognitive systems and how cognitive systems are always concerned with human like intelligence. Lastly, we talked about the overall structure of the course, which is broken up into eight large categories, like learning, planning and analogical reasoning. Next time we'll talk a little bit more specifically about this class in particular. The goals, the outcomes and the learning strategies, and what projects you'll complete.

20 - The Cognitive Connection

[Click here to watch the video](#)

At the conclusion of this lesson, we'll have a short video after the wrap up called the cognitive connection. Knowledge based AI is richly connected to cognitive signs. And so many of the topics that we'll cover in this class are connected to human reasoning and human learning,

and human memory. The cognitive connections are not separate from the course. One of the goals of this course is to learn how to use the design of AI agents to reflect on human cognition. The cognitive connections will serve this purpose.

21 - Final Quiz

[Click here to watch the video](#)

This brings us to the first quiz. After every lesson in this course we'll have a short quiz, in which we'll ask you to write down what you learned in this lesson in this blue box here. These quizzes have two goals. The first goal is to help you synthesize and organize what you have learned. The process of writing down what you learned may help you, in fact, learn it more deeply. The second goal is to provide us with feedback. Perhaps we could have been clearer or more precise about some of the concepts. Perhaps we left some misconceptions. Note that these quizzes are completely optional.

22 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

This lesson covers the high-level concepts of KBAI, Cognitive Systems, the characteristics of AI problems and AI agents, the fundamental processes of KBAI and a high-level roadmap of this course in learning the 8 major units of KBAI.

References

1. Russell, S., & Norvig, P. Artificial Intelligence: A Modern Approach, Chapter 1. Section 1.1
2. Winston, P., Artificial Intelligence: Videos and Material [Click here](#)
3. Stefk, M. Introduction to Knowledge Systems.
4. aitopics.org [Click here](#)

Optional Reading:

1. What is AI anyway? T-Square Resources ([AI-GoelDavies- Week1.pdf](#))
2. Russell & Norvig, Ch. 1 Section 1; T-Square Resources ([Russel and Norvig Ch 1-1.pdf](#))
3. The Cognitive Systems Paradigm; T-Square Resources ([Langley-paradigm - Week2.pdf](#))
4. Four views of intelligence; [Click here](#)
5. The Knowledge Level; [Click here](#)
6. An Alternative to the Turing Test; [Click here](#)

Exercises

None.

Lesson 02 - Introduction to CS7637



Perhaps the hardest truth to face, one that AI has been trying to wriggle out of for 34 years, is that there is probably no elegant, effortless way to obtain this immense knowledge base. Rather, the bulk of the effort must (at least initially) be manual entry of assertion after assertion.

– Guha and Lenat: authors of the CYC knowledge base.

01 - Preview

[Click here to watch the video](#)

Lesson Preview

- Class goals, outcomes, and strategies
- Class projects and assessments
- Computational psychometrics
- Raven's Progressive Matrices
- Principles of CS7637

commonly reoccurring principle in this class and you should be on the lookout for them.

02 - Class Goals

[Click here to watch the video](#)

You will learn...

- Core methods of knowledge-based AI
- Tasks addressed by knowledge-based AI
- How knowledge-based AI agents use these methods to address these tasks
- The relationship between AI and human cognition

Figure 26: Preview

In this lesson, we'll talk most specifically about what you should expect from CS7637. We'll start by talking about the learning goals, the learning outcomes, and the learning strategies that we'll use for this class. Then we'll discuss the class projects and assessments. That will lead us to talking about something called Computational Psychometrics, which is one of the multi-weighting principles behind the projects in this class. Next we'll talk about the Raven's Progressive Matrices test of intelligence. I think you're going to find it fascinating. The Raven's Progressive Matrices test of intelligence are the most commonly used tests of human intelligence. And that test was the target of the projects in this class. Very ambitious, you're going to enjoy it. Finally we'll discuss something

Figure 27: Class Goals

There are four major learning goals for this class. First, you'll learn about the core methods of knowledge-based AI. These methods include schemes for structured knowledge representation, methods for memory organization, methods for reasoning, methods for learning, [articulate] architectures as well as methods for meta reasoning. Meta reasoning is reasoning about reasoning. Second, you learn about some of the common tasks addressed by knowledge-based AI, such as classification, understanding, planning, explanation, diagnosis, and design. Third, you will learn ways AI agents can use these methods

to address these tasks. Fourth, you learn the relationship between the knowledge-based AI cognitive science. Using theories of human cognition to inspire their design of human level, human-like AI and using AI techniques to generate testable hypothesis about human cognition.

03 - Class Outcomes

[Click here to watch the video](#)

You will be able to...

- Design and implement a knowledge-based AI agent
- Use these strategies and agents to address complex, practical problems
- Use the design and results of these agents to reflect on human cognition

Figure 28: Class Outcomes

What are the learning outcomes of this course? At the conclusion of this class, you will be able to do three primary things. First, you'll be able to design, implement and evaluate and describe knowledge-based AIs. The design and description of knowledge-based AI agent is really the first learning goal. In order to be able to design an agent, you need knowledge of the methods of knowledge-based AI. Second, you will also be able to use these strategies to address practical problems. This learning outcome addresses the second learning goal, where you will be able to [What are the] the relationship between AI agents and real world problems. Third, you'll also be able to use the design of knowledge-based AI agents to reflect on human cognition and vice versa. This addresses the fourth learning goal.

You will complete...

- Projects
- Short assignments
- Tests
- Exercises
- Discussions

Figure 29: Class Assignments

During this course, you'll complete a variety of different kind of assessments. These assessments play different roles. First, they help you learn by demonstrating and testing what you know. Second, they help you reflect on what you've learned. Third, they help us understand what material is being taught well, and what is not being taught well. The main assessment of the projects. You'll complete a series of programming projects in designing AI agents that address a pretty complex task. We'll talk a little bit more about it in a few minutes. Second, written assignments. We'll compete a number of written assignments that will tie the course material to the projects. Third, tests. There will be two tests in this class using the content of this class to introduce a broad variety of problems. Fourth, exercises. Throughout the lessons there'll be a number of exercises to help you evaluate and manage your own learning. Fifth, interactions. We'll be looking at the interactions of the forum and other places to get a feel for how everyone is doing and how can we help improve learning

05 - Class Strategies

[Click here to watch the video](#)

You will learn by...

- Learning by Example
- Learning by Doing
- Project-Based Learning
- Personalization
- Learning by Reflection

04 - Class Assignments

[Click here to watch the video](#)

Figure 30: Class Strategies

In this class we'll use five main learning strategies. First, Learning by Example. Almost every lesson of this class starts with an example of the type of reasoning we want you to learn. The example that runs throughout the lesson, could demonstrate that reasoning. Second, Learning by Doing. In most lessons, the lesson will end with a multi-part exercise, where you are doing the exact reasoning that you learned in that lesson. There's first you see an example, then you do a similar example yourself. Third, Project-Based Learning. The class is largely structured around a series of challenging projects. And you will frequently be asked to relate each lesson you learn, to the projects in the class. Personalized learning. Personalization permeates throughout this course. You can watch the lessons in any order you choose, and at your own pace. You can choose which concepts to focus on, and everything the assignments. You'll receive personal feedback on every exercise throughout the course. Fifth, Learning by Reflection. At the conclusion of each lesson, you'll be asked to reflect on what you learned in that particular lesson. At the conclusion of each project, you'll write a designed report that will reflect on the experiments that you did as part of the project. We'll also use other learning strategies as needed, such as collaborative learning.

- Problems**
- 2×2 matrix problems
 - 3×3 matrix problems
 - 2×1 matrix problems

Figure 32: Introduction to Computational Psychometrics

Let us talk about Computational Psychometrics a little bit. Psychometrics itself is a study of human intelligence, of human aptitude, of human knowledge. Computational Psychometrics for our purposes, is the design of computational agents that can take the same kind of tests that humans do, when they are tested for intelligence or knowledge or aptitude. Imagine that you design an AI agent that can take an intelligence test. After designing it, you might want to analyze how well does it do compared to the humans on that test? You might also want to compare the errors it makes with the errors that humans make. If it does as well as humans do and if its behavior, its errors are the same as those of humans, you might conjecture then that perhaps its reasoning mirrors that of humans. In this class, we are going to be designing AI agents that can take the Raven's Test of Intelligence. In the process, we will want to use this agents to reflect on how humans might be addressing the same intelligence tests.

06 - Introduction to Computational Psychometrics

[Click here to watch the video](#)

07 - 2x1 Matrices I

[Click here to watch the video](#)

Raven's Progressive Matrices

- Test written in the 1930s to examine general intelligence.
- Consists of 60 multiple-choice visual analogy problems.
- Unique in that problems are strictly visual.
- Widespread usage as a valid test for intelligence.

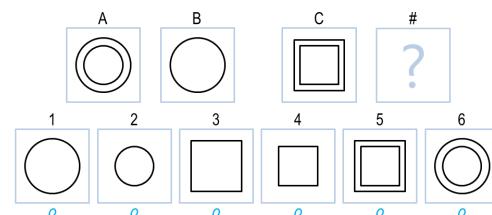


Figure 31: Introduction to Computational Psychometrics

Figure 33: 2x1 Matrices I

Let us consider an example. We are shown initially three images, A, B and C. And you have to pick a candidate for the D image here on the top right. And it can be one of these six candidates that would go here in the D image. Given that A is to B, as C is to D, what would you pick among the six choices at the bottom to put into D?

08 - 2x1 Matrices I

[Click here to watch the video](#)

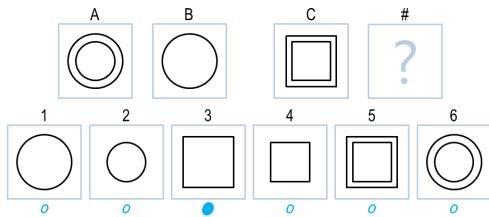


Figure 34: 2x1 Matrices I

Very good, that is in fact the correct answer for this problem. Now, of course, here's a situation where a human being, David, answered this problem. The big question for us would be, how to write a air agent that can solve this problem?

09 - 2x1 Matrices II

[Click here to watch the video](#)

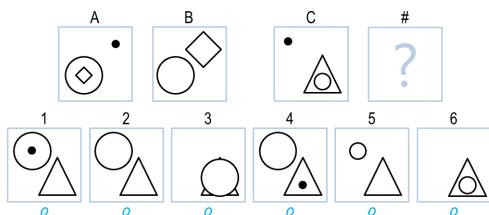


Figure 35: 2x1 Matrices II

The previous problem was pretty simple. Let's try a slightly harder problem. Once again, we're given A, B, C, and D. Given that A is to B, what would we pick between 1, 2, 3, 4, 5, and 6 to put into D?

10 - 2x1 Matrices II

[Click here to watch the video](#)

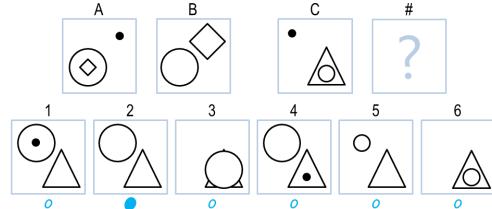


Figure 36: 2x1 Matrices II

What do you think is the right answer for this one, David? So, I observe two things are going on between A and B. The dot in the top right is disappearing and the diamond is moving out and growing. So I said that the dot would still disappear even though it's on the other side of the frame and the circle that was inside the triangle will move out and grow. So I said the answer was 2. Here I've drawn a connection between the circle and the triangle, the diamond and the circle, and the dot and its analogous dot. 2 seems to be the right answer David. Notice that the circle in 2 is on the left of the triangle even though the diamond in B, was on the right of the circle. That is okay because the diamond is replacing the dot here. And similarly, the circle is replacing the dot here. So if the dot was in the left of triangle here, it makes sense to put it in the circle and left of the triangle. Notice the drawing and connection between the diamond here, and the circle here was very easy for us to do. But designing an agent that make that kind of connection is much more difficult. So we'll deal with lots of methods, to help agents make what are very natural conclusions for us. This of course raises another issue. How do we do it? How do you solve the problem? Why was it so easy for you? Why is it so hard for A.I.? Yet another question, when David was trying to solve this problem, he looked at the relationship between A and B and then mapped it to C and some image here. But one could have gone about it the other way. One could have picked any one of these images, put it in the D and asked whether this would be a good

fit. So in one case, one can start from the problem and propose a solution. In another case, one can take one of these solutions at a time, put it here and see whether it matches. Two different strategies.

11 - 2x1 Matrices III

[Click here to watch the video](#)

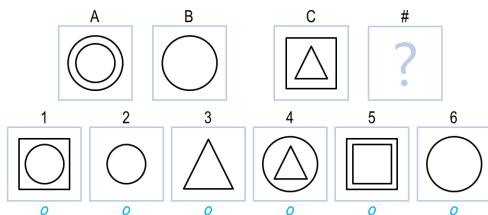


Figure 37: 2x1 Matrices III

Let's try an even harder problem. And as you solve this problem, think a little bit about how do you go about solving it.

12 - 2x1 Matrices III

[Click here to watch the video](#)

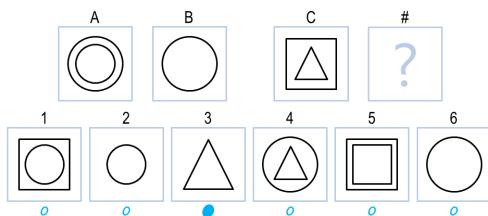


Figure 38: 2x1 Matrices III

What do you think is the correct answer, David? So on the left, we have the same two frames we had in the first problem. So first, I thought that the circle in the middle disappears, so the triangle should disappear. But none of these options match that. So then I went back and looked and said, the other way we can think about this is to say the circle on the outside disappeared but the circle on the inside grew, since their both circles we can't really tell the

difference between those, but once we know that the correct answer is not just the big square, we can say the only logical conclusion is to say that the square disappeared, and the triangle grew. So the answer has to be three, the big triangle. That's a correct answer, David. But notice something interesting here. This is an example of generate and test. You initially generated an answer from it and then tested it against the choices of a level. Yet the test failed, so you rejected a solution. And you generate another solution. For that one, the test succeeded, and you accepted it.

13 - 2x1 Matrices IV

[Click here to watch the video](#)

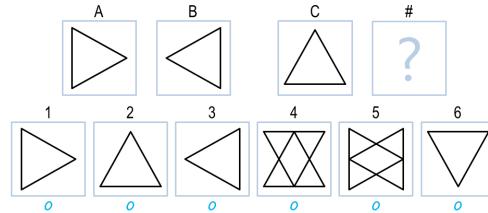


Figure 39: 2x1 Matrices IV

I like this problem. This one is really interesting. Everyone, try to solve this one.

14 - 2x1 Matrices IV

[Click here to watch the video](#)

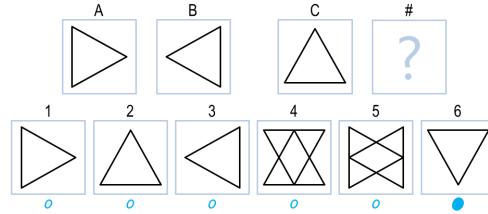


Figure 40: 2x1 Matrices IV

Okay. What do you think is the right answer to this one, David? So what I said is that it looks like there's a 180 degree rotation going on. So this frame is rotated 180 degrees to get this one. So I'm going to take C, rotate it 180

degrees to get number six. That's a fair answer, David. Well done. But notice there is another possible answer here. Two is also a possible answer. Why is two a possible answer? Because one can imagine that B is really a reflection of A across a vertical axis, and that way if we think of a vertical axis on C, then two will be the deflection/ of C on the vertical axis. So both two and six are good answers here. And one question will then become, which one do humans pick, do they pick six or do they pick two? And second which one should an AI program pick? Six or two, and how would you make sure that the AI program picks two or six, and if you are thinking I am going to give you the answer, sorry to disappoint you. I'm going to leave this as a puzzle for you. Your AI program will address this problem

15 - 2x2 Ravens Progressive Matrix I

[Click here to watch the video](#)

Is David right?

- Yes, because intelligence is more than just an algorithm for processing inputs.
 - No, because that's the same way humans operate; we just don't know how humans think yet.
 - Yes, but for another reason:
-
-

Figure 41: 2x2 Ravens Progressive Matrix I

Okay, here are some two by two problems. Two by two matrix problems. The situation is somewhat similar, but not exactly similar. Once again, we're given A, B, C, and D is unknown, and we're given six choices, 1, 2, 3, 4, 5, 6, and we are to pick one of these choices and put it in D. What is different here however is that, this time it is not just that A is to B as C is to D but also A is to C as at B is to D. That's why it's a two by two matrix. So it's not just the horizontal relationship that counts but it's also the vertical relationship that counts.

16 - 2x2 Ravens Progressive Matrix I

[Click here to watch the video](#)

Okay David, are you ready for this one? What do you think is the right answer? So I said that 3 is the right answer. Going left to right, the square became clear, so the circle becomes clear. Going top to bottom, the square becomes a circle, so the square becomes a circle. So the 3 preserve the relationships both horizontally and vertically. That's the right answer! But this was an easy problem. Let's see how you will do on a harder problem David.

17 - 2x2 Ravens Progressive Matrix II

[Click here to watch the video](#)

Principles of CS7637

1. KBAI agents represent and organize knowledge into knowledge structures to guide and support reasoning.
2. Learning in KBAI agents is often incremental.
3. Reasoning in KBAI agents is top-down as well as bottom-up.
4. KBAI agents match methods to tasks.
5. KBAI agents use heuristics to find solutions that are good enough, though not necessarily optimal.
6. KBAI agents make use of recurring patterns in the problems they solve.
7. The architecture of KBAI agents enables reasoning, learning, and memory to support and constrain each other.

Figure 42: 2x2 Ravens Progressive Matrix II

Okay, here is a slightly harder problem. Why don't we all try to solve it?

18 - 2x2 Ravens Progressive Matrix II

[Click here to watch the video](#)

Frequently-used readings

- Artificial Intelligence by Patrick Winston
- Knowledge Systems by Mark Stefik
- Artificial Intelligence by Elaine Rich and Kevin Knight
- Artificial Intelligence: A Modern Approach by Stuart Russell and Peter Norvig

Figure 43: 2x2 Ravens Progressive Matrix II

What do you think is the right answer to this one, David? So this one reminded me of that third problem we did. The first thing I thought was that it looks like the entire figure is rotating. So I'm going to say that the figure will be this with the triangle pointing up and to the left, or

up and to the right. However, looking over here, there are no answers that have the triangle rotated. So second thing I think is maybe just the outside figure is rotating. The square here rotated while the circle stayed stationary. So the circle here rotated, while the triangle stayed stationary. Because it's a circle we can't actually tell a visible difference between the two, but it seems to be the one that most preserves a relationship between A and B. Similarly, between A and C, the square on the outside becomes a circle on the outside. It's the same thing here, the square on the outside becomes a circle on the outside. That's a good answer David. Here is another point to note. Supposing we put one here in D, then C and D become identical, but A and B are not identical. Is that a problem here? Not really because we can imagine that the outer image in A is rotating to become B. And we can imagine that C in the outer image and C sort adding to become D. It just so happens that the resulting image is identical to the inchman. Note that this will be a challenge for an AI program. The AI program will have to generate solutions. It will have to evaluate solutions. [computationally] programmed solutions that it self generates. What kind of knowledge representations would it allow it to generate good solutions? What reason strategies would allow it to generate plausible answers?

20 - 2x2 Ravens Progressive Matrix III[Click here to watch the video](#)

For this one, David, what do you think is the right answer? So I put that the right answer was number 5. Number 5 looks like it preserves the reflection that's going on across the horizontal axis. That's a good answer, David, but note that 2 is also a plausible answer. One can imagine that the image A is rotating by 90 degrees to the image B. And if we rotate the image in C by 90 degrees, we'll get the answer 2. So both 5 and 2 here are plausible answers. Okay, and the question arises, which answers do most humans choose? Why do they choose the answer that they do choose? What in their cognition is telling them to choose one answer over the other? And then how can we write an AI program that can choose one answer over the other? An interesting thing to note about this problem as well is that I phrased this as reflections. Ashok defined number 2 as rotations. But it's possible that we could do this a third way. Instead of looking at rotations or reflections, which are kind of semantic ways of describing the transformations, we could look at which image completes the overall picture. Here, number 5 would seem to be the right answer, because it finishes creating the square we see forming. So that would be a strictly visual way of doing this problem as well. One more thing to note here. So far we have been talking about horizontal and vertical relationships and not diagonal relationships. So A is to B as C is to D, and A is to C as B is to D. What about diagonal relationships? Should A to D be as B to C? If we add that additional constraint, then the choice between 2 and 5 becomes clear. 5 is the right choice because that's the only way we'll get A is to D as B is to C.

19 - 2x2 Ravens Progressive Matrix III[Click here to watch the video](#)**To recap...**

- Class goals, outcomes, and strategies
- The Project: Raven's Progressive Matrices
- Principles of CS7637

21 - 3x3 Ravens Progressive Matrix I[Click here to watch the video](#)

Figure 44: 2x2 Ravens Progressive Matrix III

Let us try one more problem from this two by two set.

Now let us look at some three by three problems. This time the matrix has three rows and three columns. We are given not just A, B, and C. We are given A, B, C in the first row, D, E, F in the second row, G and H in the third row. We

do not know what will go here under I. Again, we want horizontal, vertical, and diagonal relationships. A is to B is to C, as D is to E is to F, as G is to H is to what? And similarly vertically. As well as diagonally. If we take all three of those constraints, rows, columns and diagonals, which would be the correct choice among one through six to put under the square?

22 - 3x3 Ravens Progressive Matrix I
[Click here to watch the video](#)

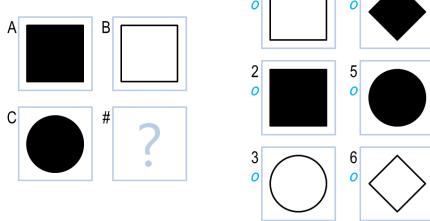


Figure 45: 3x3 Ravens Progressive Matrix I

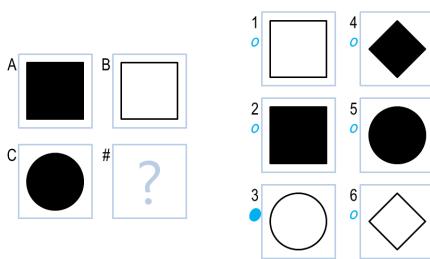


Figure 46: 3x3 Ravens Progressive Matrix I

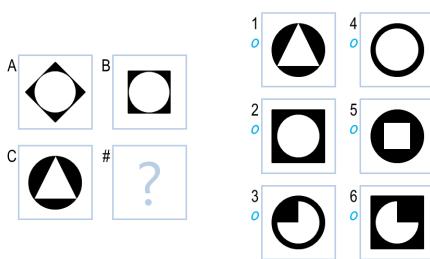


Figure 47: 3x3 Ravens Progressive Matrix I

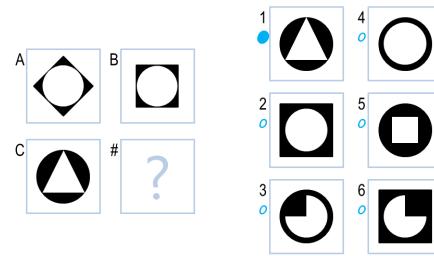


Figure 48: 3x3 Ravens Progressive Matrix I

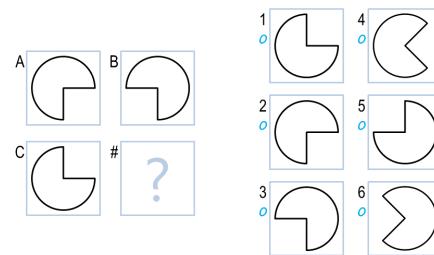


Figure 49: 3x3 Ravens Progressive Matrix I

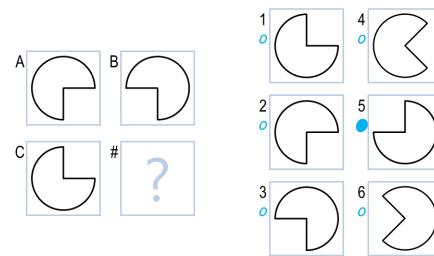


Figure 50: 3x3 Ravens Progressive Matrix I

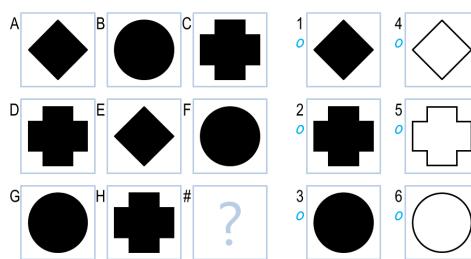


Figure 51: 3x3 Ravens Progressive Matrix I

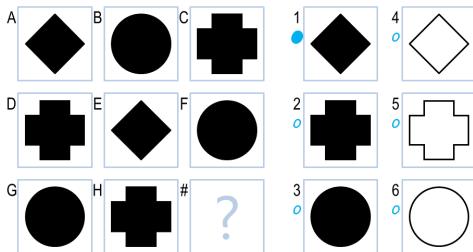


Figure 52: 3x3 Ravens Progressive Matrix I

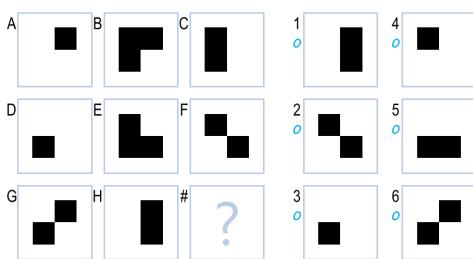


Figure 53: 3x3 Ravens Progressive Matrix I

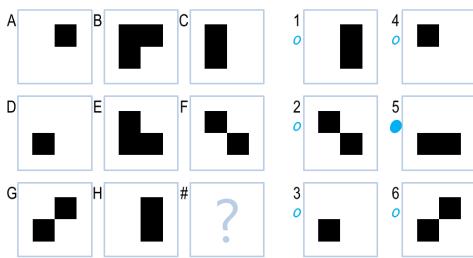


Figure 54: 3x3 Ravens Progressive Matrix I

What do you think is the right answer David? So, looking horizontally every row except for the third row has a diamond. Vertically every column except the third column has a diamond. And diagonally the shapes are preserved if we imagine C'd be coming down here, G coming up here. So it seems like all signs point to number one. That is indeed the correct answer. But, you said something very important about this particular problem. And that is that, we can imagine that these rows are rotating, so that this C gets aligned with D, and D gets aligned with H, as if they were on a diagonal. One more point about this. Once again David was able to solve this problem within a few seconds. What

about an AI program? How could the AI program solve this problem? What representations would it use? What reason strategies would it use? Would it induce something from the first row? Would it learn something from the first row and apply it to the second row? If so, how would it do that induction?

23 - 3x3 Ravens Progressive Matrix II

[Click here to watch the video](#)

Okay, let's try a harder one. I can tell you that this problem is hard. Even I have difficulty with this problem. So let's take a minute, think about it. Again, this is a three by three matrix problem. You're given the six choices to the right.

24 - 3x3 Ravens Progressive Matrix II

[Click here to watch the video](#)

Okay, David, are you ready for this one? What answer did you come up with? So after pondering it for far too long, I finally came to the answer that the answer is five. This problem is very different than the one that we've seen in the past, because it's the relationship between the first two frames in each row and column that dictates the third frame. The relationship is called exclusive or. If the box appears in both the first two frames in a row or column, it doesn't appear in the third one. If it appears in neither of the first two frames, it doesn't appear in the third one, but if it appears in exactly one of the frames, it appears in the third one as well. So here the top right square appears in both A and B, so it doesn't appear in C. The top left and bottom left squares appear in B only, so they do appear in C. If you look across the rows and down the columns, you'll see that, that relationship holds true for every row and column. And in fact, both rows and columns give us five as the answer. So if the row, the bottom left and bottom right appear each in only one of those frames, while the top right appears in both. So bottom left and bottom right appear here. For the right column, top left and top right appear both times, while bottom left and bottom right each only appear once. So the answer here again

is bottom left and bottom right. That was excellent. Five looks like the right answer. Now did you follow the same strategies this time that you had followed the last time? No, definitely not. In the earlier problem, we saw that the first row had relationships that carried through for every single row. It didn't really matter what order the figures were in. All that mattered was the relationships between them. Here, if we were to switch around some of the figures it would change what the other figures would have to be. And change the nature of the relationship inside each row and column. So we've used a fundamentally different kind of reasoning process. And that's part of what makes this problem so difficult is that it's unlike the ones we've seen in the past. That's very interesting, couple of other things to note as well. I wonder whether this time you actually pick one, put it under here, and then solve with the completed pattern or not. And if they did not succeed, you pick the second one and put it here and solve the patterns succeeded or not and went through it systematically. And finally came up with five because that fit the pattern the best. If that were the case, then this would be a different strategy from looking at the first row and the second row and the first column and the second column, inducing some rule and applying it with the third row. One other thing to note, something very interesting about knowledge based AI. We can ask ourselves, how is David solving these problems, if we can figure out, if we can generate hypothesis about how David is solving these problems, then that will inform how we can build an AI agent that can solve these problems. There is a side where we're going from human cognition to AI. Alternatively, we can write an AI program that can solve these problems and by looking at how we are programmed to solve these problems we can generate hypotheses about how David might be solving problems. That's going from the AI side to the cognitive science side.

26 - Exercise What is intelligence

[Click here to watch the video](#)

If you are designing a AI agent that can take an intelligence test. That raises the question, if

we succeed is the AI intelligent. What do you think, David? So, I would say no, even if the agents that we design successfully solve the intelligence test, they aren't themselves intelligent, they are just processing signals and inputs in the correct way. What do you think?

27 - Exercise What is intelligence

[Click here to watch the video](#)

The problem with David's answer, in my opinion is that at a certain level humans too are just processing signals and inputs in the right way. What then makes us intelligent? Intelligence is hard to define. In the life sciences, scientists study life, but don't always agree on a definition of life. Similarly, in the cognitive sciences, we study intelligence, but don't necessarily define it. And knowledge based AI, will take the view that knowledge is central to human level intelligence.

28 - Principles of CS7637

[Click here to watch the video](#)

Our discussion of knowledge-based agents in this CS7637 class is organized around seven principals. Be on the lookout for the seven principals, they'll occur again and again throughout the course. Principal number one, agents use knowledge to guide reasoning and they represent and organize this knowledge into knowledge structures. Principle number two, learning is often incremental. This connects back to one of the characteristics of problems, where data and experience was coming incrementally. Number three, reasoning is top down not just bottom up. We just don't reason from data, we also use data to pull out knowledge from memory, then we use this knowledge to generate expectations to make sense of the world. Principle number four, knowledge-based AI agents match methods to tasks. We'll discuss a number of different methods. We'll also discuss several tasks. Later, we'll discuss how AI agents can select specific methods to address particular tasks and even integrate different methods to address complex tasks. Principle number five, agents use heuristics to find solutions that are good enough. They do not necessarily find optimal solutions. This

is because of a trade off between computer efficiency and the optimality of solutions. Our focus will be on using bounded rationality and yet, giving near real-time time performance on [AI-though you will be] intractable problems. This happens, because agents use heuristics to find solutions that are just good enough. Principle number six, agents make use of recurring patterns of problems in the world. Of course, there are a number of problems. But even this number of problems are characterized the patterns that occur again and again. Number seven, reasoning, learning and memory constrain and support each other. We'll build theories that are not just theories of reasoning or theories of learning or theories of memory, but we'll build theories that unify the three of them into one single cognitive system. These principles will come again and again in this particular class. So we highly recommend that you take a moment, pause the video and read over the principles once again.

29 - Readings

[Click here to watch the video](#)

Throughout this course, we will be using materials drawn from a number of different textbooks and papers. You'll find specific references to all these sources for this examples in the Class Notes. Generally, however, we will use a handful of books from which we will draw a lot of material. Artificial Intelligence by Patrick Winston. Knowledge Systems by Mark Stefik, Artificial Intelligence by Elaine Rich and Kevin Knight and Artificial Intelligence: A Modern Approach by Stuart Russell and Peter Norvig. You don't have to buy any of these books. But if you want to increase your knowledge beyond what we'll discuss in this class, you may want to consider both these text books and the specific references at the end of each lesson.

30 - Wrap Up

[Click here to watch the video](#)

Today we started off by discussing the goals, outcomes, and learning strategies for this class. That led us into discussing the project, the main assessment in this class. The project builds

on Raven's Progressive Matrices, an apocryphal human intelligence test, an idea called computational psychometrics, the application of computer models to understanding human cognition. We then discuss the seven main principles of CS7637 that are going to come up again and again in this course. We recommend you keep and eye out for them because they'll come up in every single lesson. This is the end of the introduction to this course, so we hope you're excited to get started. We're going to start off with the fundamentals of knowledge based AI, including one of the most fundamental knowledge structures called semantic networks.

31 - The Cognitive Connection

[Click here to watch the video](#)

Let us look at a connection between your class projects and human cognition. Beginning with psychometrics, psychometrics is the science of measuring human intelligence, aptitude, and knowledge. Computational psychometrics is the science of building agents that can take the same tests of intelligence that humans take. All the [computational] building AI agents that can address the Ravens test of intelligence, it will provide opportunities for thinking about human cognition. While we will be looking only for how well your agents perform on the Ravens test, in principle [computational] psycho-metrics will also look at the kinds of errors that AI agents make. If the errors the AI agents make are similar to those that humans make, then that may provide a source of hypothesis about human thinking on this Raven's Test of Intelligence. It is also interesting to note that people with autism perform about as well on the Raven's Test of Intelligence as neurotypical people. This is a little surprising, because in general, people with autism do not perform as well on other tests of intelligence as the neurotypical people. Note that Raven's test of intelligence is the only test that consists only of visual analogy problems. All other tests of intelligence also include a large number of verbal problems. Might I suggest that some of the thinking strategies that people with autism are better aligned with visual reasoning.

32 - Final Quiz

[Click here to watch the video](#)

And now to the quiz at the end of this lesson. Will you please fill out what you learned in this lesson?

33 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

This lesson cover the following topics:

1. Relationship between AI and human cognition,
2. Use of Raven's progressive matrices to understand human cognition,
3. Generalizing knowledge to solve problems with similar patterns,
4. Use of human intelligence tests could be used to understand brain development in autistic

people and may be someday treat them:

[Click here](#)

5. Use human cognition to build better AI agents and vice-versa.

References

1. Russell, S., & Norvig, P. Artificial Intelligence: A Modern Approach, Chapter 1. Section 1.1

Optional Reading:

1. Putting Online Learning and Learning Sciences Together; [Click here](#)
2. Understanding the Natural and Artificial Worlds; [Click here](#)

Exercises

None.

Lesson 03 - Semantic Networks



I think most people can learn a lot more than they think they can. They sell themselves short without trying. One bit of advice: It is important to view knowledge as a sort of a semantic tree – make sure you understand the fundamental principles, i.e. the trunk and big branches, before you get into the leaves/details or there is nothing for them to hang on to.

– Elon Musk: co-founder of PayPal, Tesla Motors; co-chairman of OpenAI.

01 - Preview

[Click here to watch the video](#)

Lesson Preview

- Knowledge representations
- Semantic networks
- Problem-solving with semantic networks
- Represent & Reason

Figure 55: Preview

Fundamentals

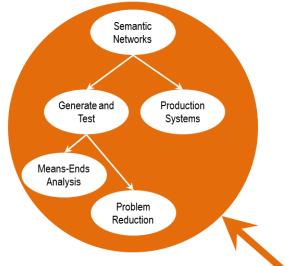


Figure 56: Preview

knowledge representations, then we'll focus on semantic networks. We'll illustrate how semantic networks can be used to address two by one matrix problems. You can think of this like a represent and reason modality. Represent the knowledge, represent the problem, then use that knowledge to address the problem. As simple as that. At the end, we'll close this lesson by connecting this topic with human cognition and with modern research in AI.

02 - Representations

[Click here to watch the video](#)

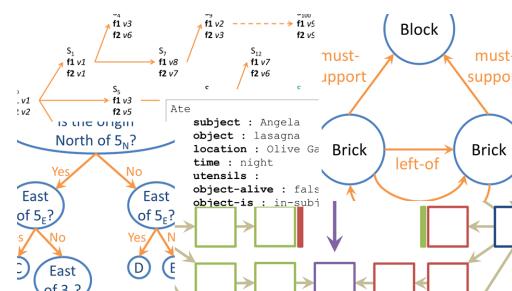


Figure 57: Representations

Okay. Let's get started with Knowledge PCI. Today we'll talk about semantic networks. This is the kind of knowledge representation scheme. This is the first lesson in our fundamental topics part of the course. We'll start talking about

So what is a knowledge representation? The collage on this screen shows several knowledge representations that AI has developed. In each knowledge representation there is a language.

That language has a vocabulary. Then, in addition to that language, in that knowledge representation there's some content. The content of some knowledge. Let's take an example with which you're probably already familiar. Consider the Newton's second law of motion. So I can represent it as $f = m \cdot a$. This is a knowledge representation. A very simple knowledge representation, in which there are two things. There is a language of algebraic equations, $y = b \cdot x$, for example. And then there is the content of our knowledge of Newton's second law of motion, force equals mass times acceleration. So the knowledge representation has two things, once again. The language, which has its vocabulary. For example, this sign of equality. And the content that goes into that representation expressed in that language. Let us not worry too much about all of the representations in this collage right now, we'll get to them later. The idea here is to simply show that AI has developed not one, but many knowledge representations. Each representation has its own affordances and its own constraints.

03 - Introduction to Semantic Networks

[Click here to watch the video](#)

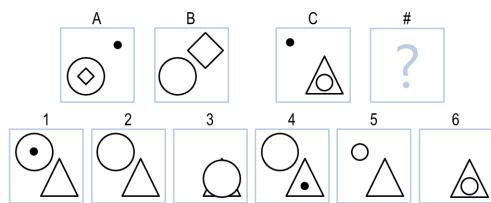


Figure 58: Introduction to Semantic Networks

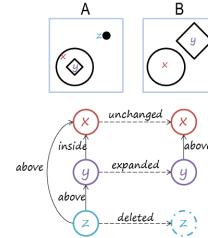


Figure 59: Introduction to Semantic Networks

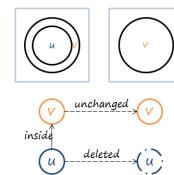


Figure 60: Introduction to Semantic Networks

To understand semantic networks as a knowledge representation, let us take an example. This is an example that we saw in a previous lesson. This is A is to B as C is to D, and we have to pick one of the six choices at the bottom that will go in D. How will we represent our knowledge of A, B, C and the six choices at the bottom? Let us begin with A and B. We'll try to build semantics networks that can represent our knowledge of A and B. Inside A is a circle, I'll label it x. Also inside x is a diamond, I'll label it y. Here is a black dot, I'll label it z. We can similarly label the objects in B. So inside A are three objects, x, y, and z. So the first thing we need to do in order to build a semantic network for representing our knowledge of A is to represent the object. So I have the object x, the object y, the object z, standing for the circle, the diamond and the black dot. Now that we have represented the objects in A, we want to represent the relationships between these objects. So I have the objects x, y, z, and we'll try to represent the relationship between them by having links between the nodes representing the objects. These links can be labeled. So I may say that y is inside x because that is the relationship in the image A. Similarly

I may say that z is above y because z is above y in the image A. I may also say that z is above x because z is above x in image A. In this way, a semantic network representation of the image A captures both the objects and the relationship between the objects. We can do exactly the same thing for the image B. The objects and the relationships between them, y is above x. Now that we have represented our knowledge of image A and our knowledge of image B we want to capture somehow the knowledge of the transformation from A to B because recall, A is to B as C is to D. So we want to capture the relationship between A and B. The transformation from A to B. To do that, to capture the transformation from A to B, we'll start building links between the objects in A and the objects in B. Now, for x and y they are straightforward for z, but there is no z in b. So we'll have a dummy node here in b and we will see how we can label the link here so that we can capture the idea that z doesn't occur in B. So we might say that x is unchanged because x the circle here is the same as the circle here. Y on the other hand has expanded. It was a small diamond here and it's a much bigger diamond there. Z, the black dot, has disappeared all together, so, we have, let's say, it's deleted, it's not there at B at all. I hope you can see from this example how we constructed a semantic network for the relationship between the images A and B. There were three parts to it. The first part dealt with the objects in A and the object in B. The second dealt with the relationships between the objects in A and the relationships with the objects in B. The third party dealt with the relationships between the objects in A and the relationships between the objects in B. In principle, we can construct semantic networks for much more complicated images, not just A and B. Here is another example of a semantic network for another set of images. Once again, we have the objects and the relationships. And then the relationship between the objects in the first image and that in the second image.

04 - Exercise Constructing Semantic Nets I
[Click here to watch the video](#)

Exercise: Write the relationships between the pieces in the blanks on the right.

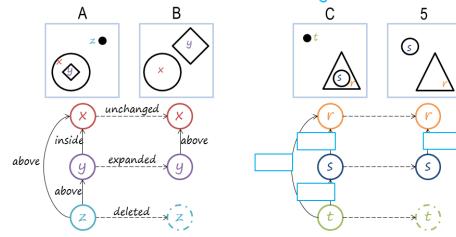


Figure 61: Exercise Constructing Semantic Nets I

Okay, very good. Here is C and I've just chosen one of the choices out of the six choices, five here. And so we're going to try to build a semantic network for C and five, just the way we built it for A and B. So for C and five, I have already shown all the objects. Now, your task is to come up with the labels with the links that are between these objects here, as well as labels for the link in the, between the object for five.

05 - Exercise Constructing Semantic Nets I
[Click here to watch the video](#)

Exercise: Write the relationships between the pieces in the blanks on the right.

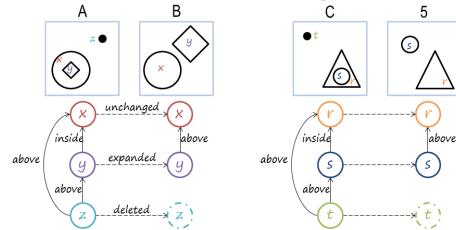


Figure 62: Exercise Constructing Semantic Nets I

Now David made an important point here. He said that the vocabulary he's using here of inside and above, is the same as the vocabulary that I had used of inside and above here. And that's a good point because we want to have a consistent vocabulary throughout the representation for the class of problems. So here we have decided that for representings, problems of this kind in semantic networks, we will use a vocabulary of inside and above and we will try to use it consistently.

06 - Exercise Constructing Semantic Nets II[Click here to watch the video](#)

Exercise: Write the transformations between the frames in the blanks on the right.

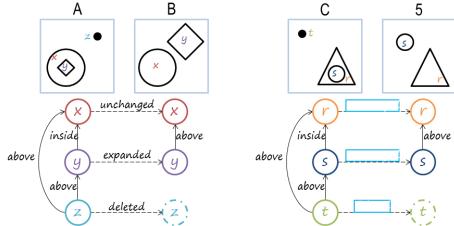


Figure 63: Exercise Constructing Semantic Nets II

Let's go one step further. Now we have the semantic network for C, and the semantic network for 5. But we have yet to capture the knowledge of the transformation from C to 5. So we have to label the, these three links.

07 - Exercise Constructing Semantic Nets II[Click here to watch the video](#)

The relationships between the pieces and the transformations between the frames.

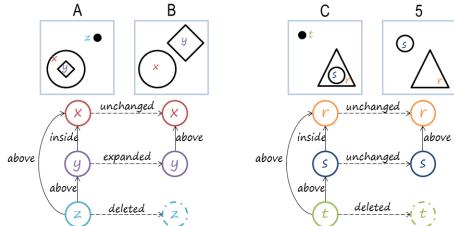


Figure 64: Exercise Constructing Semantic Nets II

Let's do this exercise together. Derrick, what labels did you come up with? So just like I tried to transfer the vocabulary we used to describe the relationships between shapes and a figure, I decided to try and transfer the vocabulary we used to describe transformations between the two figures. So just like X was unchanged between A and B, r is unchanged between C and five. In this case, s is also unchanged. It moved, but that's captured in the relationship between the shapes in figure five. The shape itself hasn't

changed, so it's unchanged. Just like dot z disappeared, dot t also disappeared so I marked it deleted like we did before. Good, that seems like a good answer.

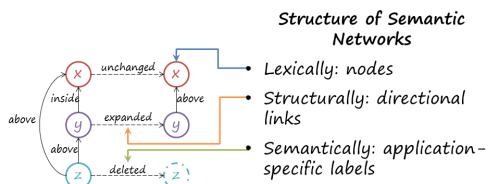
08 - Structure of Semantic Networks[Click here to watch the video](#)

Figure 65: Structure of Semantic Networks

Now that we have seen some examples of semantic networks, let us try to characterize semantic networks as a knowledgeable presentation. A knowledgeable presentation, will have a lexicon. That tells us something about the vocabulary of the presentation language. A structure which tells us about how the words of that vocabulary can be composed together into complex representations and the semantics which tells us how the representation allows us to draw inferences so that we can in fact reason. In case of semantic network the basic lexicon consists of nodes that capture objects. So, x, y, z. What about this structural specification? Structural specification here consists of links which have directions. These links capture relationships and allows to compose these notes together into complex representations. What about the semantics? In case of semantics, we are going to put labels on these links which are then going to allow us to do, draw inferences and do reasoning over these representations.

09 - Characteristics of Good Representations[Click here to watch the video](#)

Characteristics of Good Representations

- Make relationships explicit
- Expose natural constraints
- Bring objects and relations together
- Exclude extraneous details
- Transparent, concise, complete, fast, computable

Figure 66: Characteristics of Good Representations

Now that we have seen semantic networks in action, we can ask ourselves the important question. What makes a knowledge representation, a good representation? Well, a good knowledge representation makes relationships explicit. So in the two by one matrix problem, there were all these objects, circles and triangles and dots. There were there this relationship between them, left off and inside. And the semantic that worked made all of them explicit. It exposed the natural constraints of the problem. A good representation works at the right level of abstraction. So that it captures everything that needs to be captured, and yet. Removes all the details that are not needed. So representation, a good representation, is transparent, concise, captures only what is needed, but complete, captures everything that is needed. It is fast, because it doesn't have all the details that are not needed. And it is computable. It allows you to draw from the inferences that need to be drawn, in order to address the problem at hand.

10 - Discussion Good Representations

[Click here to watch the video](#)

Is David right?

- Yes, because it lists all nutritional information explicitly.
- Yes, because it only provides the information relevant to the consumer.
- No, because it does not make explicit connections between linked variables like fat and fat calories.
- No, because it is not computable by a computer.

Figure 67: Discussion Good Representations

So, what's a good representation for everyday life? David, can you think of an example? So a good example that comes to mind pretty quickly for me is the example of the nutritional information on the side of a food or beverage container. It lists slots for each different kind of nutrient, and also lists the quantity that's present in that food. So it seems like it's a pretty good representation of what's actually going on inside the food container. What do you think? Do you think David is right?

11 - Discussion Good Representations

[Click here to watch the video](#)

Is David right?

- Yes, because it lists all nutritional information explicitly.
- Yes, because it only provides the information relevant to the consumer.
- No, because it does not make explicit connections between linked variables like fat and fat calories.
- No, because it is not computable by a computer.

Figure 68: Discussion Good Representations

Let us think through this a little bit more deeply. Does the nutritional label make all the information explicit, David? So it seems like it does, for each given nutrient in the food, it lists the explicit amount of that, that is present in the food. So it seems like it makes all the information explicit. Would you say then, it enables me to draw the right kind of inferences? Well, it certainly seems like it does. So, if I'm worried about my diet and I'm worried about how many calories I'm consuming, it tells me right there on the label exactly how many calories it has. So, I can infer the right amount of food to eat per day. If you're concerned with something different in your diet, though, it actually gives you the information to make a different kind of inference. What about extraneous information, information that I may not need? Well, that's a good point, actually. Because it gives the information that anyone might need and, as a result, it's going to give information that's extraneous to somebody. So, for example, in my diet,

I might not care about the number of carbohydrates I consume. So, for me, that's extraneous information. But, for somebody else, that's very important. So it doesn't avoid extraneous information, but it would be really hard for it to do so. What about relationships? Does it make all the concepts of all the relationships between them really explicit? So actually, now that you mention it, Ashoke, it really doesn't. So for example, the number of calories from fat in a particular food is based, in large part, on the amount of fat that's in it. It gives the number of calories from fat and the amount of fat in it right there on the label, but it doesn't tell me that there's a relationship there. So I really can't make any inferences based on that relationship. I don't even know that relationship exists. So note how this connects with the ability to make inferences. Nutritional labels capture some information that allows us to make good inferences, do not capture all the information.

12 - Guards and Prisoners

[Click here to watch the video](#)

Guards & Prisoners Problem

- Also known by other names (cannibals and missionaries, jealous husbands, brothers and sisters).
- Originally appeared in the 1200-year-old text *Propositiones ad Acuendos Juvenes*.
- Used by throughout AI for problem representation.



Figure 69: Guards and Prisoners

Guards & Prisoners Problem

- Three guards and three prisoners must cross river.
- Boat may take only one or two people at a time.
- Prisoners may never outnumber guards on either coast, though prisoners may be alone on either coast.

Figure 70: Guards and Prisoners

Let us now look at a different problem, not a 2 by 1 matrix problem but a problem called

the guards and prisoners problem. Actually this problem goes by many names, Cannibals and missionaries problem, the jealous husbands problem and so on. It was first seen in a math text book about 880 and has been used by many people in AI for discussing problem representation. Imagine that there are three guards and three prisoners, on one bank of the river and they must all cross to the other bank. There is one boat, just one boat and they can only take one or two people at a time, not more and the boat cannot travel alone. On either bank, prisoners can never outnumber the guards, if they do they will overpower the guards. So, the number of guards must at least be equal to the number of prisoners on each bank. We'll assume these are good prisoners. They won't runaway if they're left alone. Although they might beat up the guards if they outnumber them. That's the beauty of this class. We lead with real problems, practical problems. We also make up problems to help illustrate specific things. I think you're going to have fun with this one.

13 - Semantic Networks for Guards Prisoners

[Click here to watch the video](#)

Figure 71: Semantic Networks for Guards Prisoners

Let us try to construct a semantic network representation, for this guards and prisoners problem, and see how we can use it to, do the problem solving. So in this representation, I'm going to say that each node is a state in the problem solving. In this particular state, there happens to be one guard and one prisoner on the left side. The boat is on the right side, and two of the prisoners and two of the guards are also on the right side. So this is a node, one single

node. So the node captured, the lexicon of the semantic network. Now, we'll add the structural part. And the structural part has to do with the transformation. That is going connect different nodes, into a more complex sentence. We'll label the links between the nodes, and these labels then, will capture some of the semantics of this representation, that will allow us to make interesting inferences, when it comes time to do the problem solving. Here is a second node, and this node represents a different state in the problem solving. In this case, there are two guards and two prisoners on the left side. The boat is also on the left side. There is one guard and one prisoner on the right side. So this now, is a, semantic network. A node, another node, a link between them and the link is labelled. Note that in this representation, I used icons to represent objects, as well as icons to represent labels of the links between the nodes. This is perfectly valid. You don't have to use words. You can use icons, as long as you're capturing the nodes and the objects inside each state, as well as the labels on the links between the different nodes.

14 - Solving the Guards and Prisoners Problem

[Click here to watch the video](#)

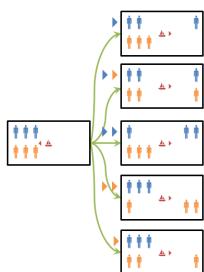


Figure 72: Solving the Guards and Prisoners Problem

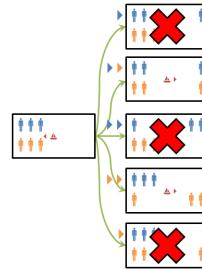


Figure 73: Solving the Guards and Prisoners Problem

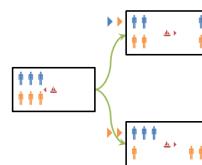


Figure 74: Solving the Guards and Prisoners Problem

There's an old saying in AI, which goes like, if you have the right knowledge representation, problem solving becomes very easy. Let's see whether that also works here. We now have a knowledge representation for this problem of guards and prisoners. Does this knowledge representation immediately afford effective problem solving? So, here we are in the first node, the first state. There are three guards and three prisoners in the boat, all in the left-hand side. Let us see what moves are possible from this initial state. Now, using this representation, we can quickly figure out that there are five possible moves from the initial state. And the first move, we move only guard to the right. On the second move, we move a guard and a prisoner to the right. In the third move, we can move two guards, or two prisoners. Or, in the fifth move, just one prisoner to the right. Five possible moves. Of course, we know that some of these moves are illegal and some of them are likely to be not very productive. Will the semantic network allow us to make inferences about which moves are productive and which moves are not

productive? Let's see further. So, let's look at the legal moves first. So we can immediately make out from this representation, that the first move is not legal because we are not allowed to have more prisoners than guards on one side, of the river. Similarly, we know that the third move is illegal for the same reason. So, we can immediately rule out the first and the third moves. The fifth move, too, can be ruled out. Let's see how. We have one prisoner on the other side. But the only way to go back would be to take the prisoner to the, back to the previous side. And if we do that, we reach the initial state. So we did not make any forward progress. Therefore, we can rule out this move as well. This leaves us with two possible moves that are both legal and productive. The, we have already removed the moves that were not legal and not productive. Later, we will see how AI programs can use various methods to figure out what moves are productive and what moves are unproductive. For the time being, let's go along with our problem solving.

15 - Exercise Guards and Prisoners I

[Click here to watch the video](#)

Write the number of guards on the left coast in the top left box, just as a number zero, one, two, or three. The number of prisoners on the left coast in the bottom left box, the number of guards on the right coast in the top right box, and the number of prisoners on the right coast in the bottom right box.

16 - Exercise Guards and Prisoners I

[Click here to watch the video](#)

So what answers did you come up with, David? So we knew that because the boat is traveling from the right back to the left, that the boat is going to be on the left in each of the next states, so we didn't even ask for that. In the top state, the only things we could do are move both people back, just the guard back, or just the prisoner back. Moving both back would just take us back to our original state, so while it's legal, it's not productive. Moving just our prisoner back would have more prisoners on the left than

guards, so that wouldn't be a legal move. So our only move is to move our guard back to the left. Similarly on the bottom, we could either move both prisoners back to the left, or just one. Moving both would take us back to our original state which isn't productive. Moving just one, then, is our only legal next state. That was a good answer, David. Thank you. Note David, that these two states that he came up with, the one at the top here and one at the bottom here, are in fact identical. And because they're identical, therefore we can collapse them into one state. So, now we get the representation shown in this figure, the two states have been collapsed into one. So in this semantic network, we don't really care how we got into a state, just as long as we know what state we're in. And that makes sense in this problem solving process. Once we're in this state, we don't care if we got to it this way, or this way, all we care about is the current state of the problem.

17 - Exercise Guards and Prisoners II

[Click here to watch the video](#)

Let us take this problem solving a little bit further. Now that we're in this state, let us write down all the legal moves that can follow. It will turn out that some of these legal moves will be unproductive, but first, let's just write down the legal moves that can follow from here.

18 - Exercise Guards and Prisoners II

[Click here to watch the video](#)

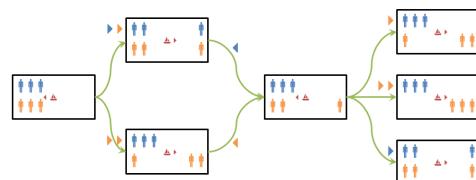


Figure 75: Exercise Guards and Prisoners II

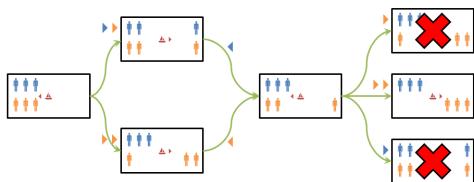


Figure 76: Exercise Guards and Prisoners II

on my own, it took me a while because I didn't recognize the fact that I kept going around in a circle because I didn't realize I kept coming back to the exact same state. In fact, David, most of us have the same difficulty. So the power of this semantic network as a representation is arising because it allows us to systematically solve this problem, because it makes all the constraints, all the objects, all the relationships, all the moves, very explicit.

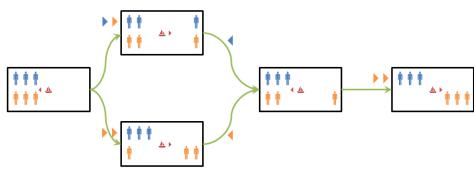


Figure 77: Exercise Guards and Prisoners II

David, did you solve this problem? So just like the original state, there could be five moves that come out of this. Two of the moves we can already say are illegal. Moving two guards would have more prisoners than guards on the left. And moving one guard and one prisoner would end up with too many prisoners on the right. So we don't include those states. Then our three legal moves are to move one prisoner, two prisoners, or one guard. But, now I notice that this state is actually identical to this state. Like we said before, once we're in a state, we don't really care how we got there. So going back to an earlier state is not a productive move. So, we can rule this out as an unproductive move. Similarly, this state is the same as this state. So we can rule this out as an unproductive move as well. That leaves us with only one legal productive move that can follow from the previous state. That is a very good point, David. So this representation is a good representation for this problem, because it is making all the constraints of the problem solving explicit. So that we can quickly compare states and say, which moves are productive and moves are not productive? I can tell you that when I tried to solve this problem

19 - Exercise Guards and Prisoners III

[Click here to watch the video](#)

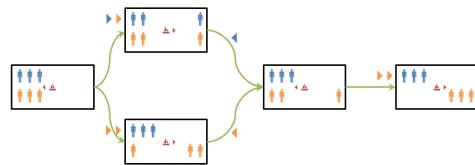


Figure 78: Exercise Guards and Prisoners III

We can continue this problem solving process further and solve this guards and prisoners problem. I'll not do that here, both because it will take a long time, and because the entire picture will not fit into the screen. But I would like you to do it yourself. And I want you to do it and tell me how many moves does it take to move all the guards and all the prisoners from one side to the other side of the river? Once you are done. Once you have solved the problem and moved all the guards and prisoners to the other side, write the number of moves here in this box.

20 - Exercise Guards and Prisoners III

[Click here to watch the video](#)

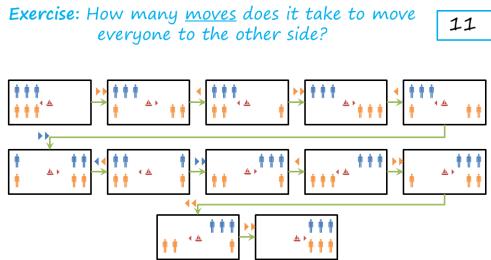


Figure 79: Exercise Guards and Prisoners III

How many moves did you come up with, David? So, I was able to do the problem in 11 moves. I won't talk you through the entire process, but if you'd like to pause this video you can take a look and see the exact moves I used, and see if they match yours. We know that 11 moves are the most efficient way to solve the problem, but these aren't the only 11 moves that can actually solve the problem. Good, David. That's the correct answer. So now that we have solved this problem together, next time you go to a cocktail party, you can entertain all of your guests by showing how you can do this problem so effectively and so efficiently. Even if all of them flounder trying to do things on their napkin that's not succeeding. So we've not yet talked about how an AI method can determine which states are productive and which states are unproductive. We'll revisit this issue in the next couple lessons.

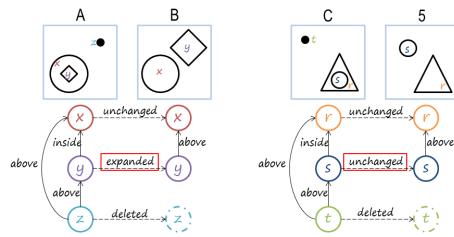


Figure 81: Represent Reason for Analogy Problems

Now that we have seen how, the semantic network knowledge representation, enables problem solving, let us return to that earlier problem that we were talking about. The problem of A is to B, as C is to 5. Recall that we have worked out the representations for both A is to B and C is to 5. The question now becomes, whether we can use this representation, to decide whether or not 5 is the correct answer. If we look at the two representations in detail, then we see part of the representation here, is the same as the representation here. Except that, this part is different from this part. Here we have y expanded and right here, we have s remain unchanged. So this may not be the best answer. Perhaps there is a better answer. Where the representation on the left, will exactly match representation on the right.

21 - Represent Reason for Analogy Problems

[Click here to watch the video](#)

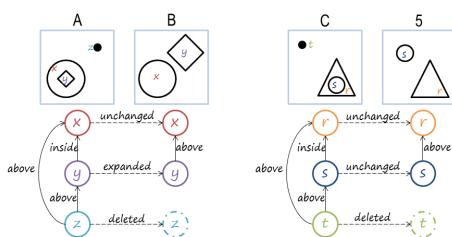


Figure 80: Represent Reason for Analogy Problems

22 - Exercise Represent Reason for Ravens

[Click here to watch the video](#)

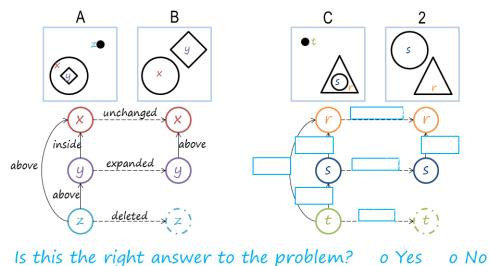


Figure 82: Exercise Represent Reason for Ravens

So, let us do another exercise together. This time I have picked a different choice, choice 2 here. So now, we can build a representation for

A is to B like earlier, and here is a representation of C is to 2. I would like you to fill out these boxes for the labels on the links here. And then answer whether or not two is the right answer for this problem.

23 - Exercise Represent Reason for Ravens

[Click here to watch the video](#)

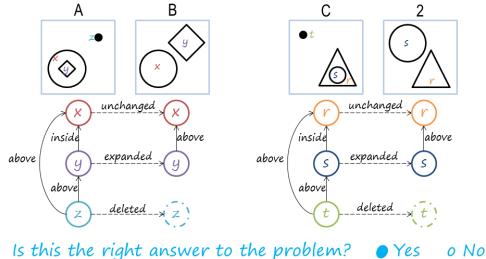


Figure 83: Exercise Represent Reason for Ravens

What answer did you come up with, David? Is two the correct answer? So I said that that was the correct answer. It looks to me like the relationships at play in frame A and frame C are the same. And the relationships at play in frame B and frame 2 are the same. We saw that x was unchanged between the two frames, just as r is. Y expanded, just s does. And z was deleted, just as t is. Similarly, y was above x, just as s is above r. So I think it's the right answer. Good David, that's in fact the correct answer. Two is the best choice of a level here. But imagine that for 2 s row, the match between the representation on the left side and the representation on the right side was only partial. Just like it was when we had five over here. How then would we make a choice between two and five, David? So it sounds like so far we've been talking about an all-or-nothing match of transformations. But if we didn't have an exact match between the transformations, we'd have to come up with some way to evaluate how well two transformations actually fit together. So we might say that these transformations are 95

24 - Exercise How do we choose a match

[Click here to watch the video](#)

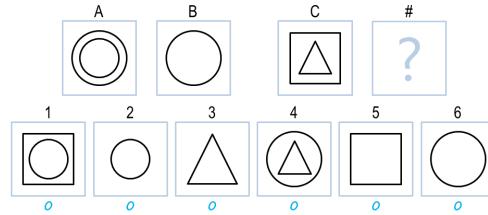


Figure 84: Exercise How do we choose a match

Let us do another exercise. This is actually an exercise we've come across earlier, however this exercise has an interesting property. Often the world presents input to us, for which there is no one single right answer. Instead, there are multiple answers that could be right. The world is ambiguous. So, here we again have A is to B, C is to D, and we have six choices. So, what choice do you think is the right choice here?

25 - Exercise How do we choose a match

[Click here to watch the video](#)

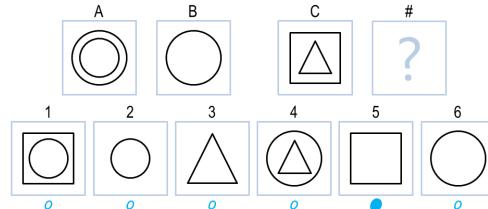


Figure 85: Exercise How do we choose a match

What answer did you think about, David? So this is a tiny bit different than the one we saw earlier. Earlier, we saw this exercise without the square as an option, so we were forced to pick the triangle. But here, since we actually have the square as the option, I think the easier transformation is to say that the middle shape is disappearing. Thus, the triangle disappears and leaves us with just that square. So, I'd say that five is the correct answer. That's a good answer, David, but note that three could also be an answer because one could imagine the relationship between A and B a little differently. Suppose we

were to say that the outer circle is disappearing and the inner circle is expanding. In that case, we will again get B. So in that case, we might say that the outer image is disappearing, this square, and the inner shape is expanding to give us three. So three is also a legitimate answer. So I imagine though, that most people would choose answer five over answer three. But why would most of us choose answer five over answer three? That's a great question. Let's look at this in more detail.

26 - Choosing Matches by Weights

[Click here to watch the video](#)

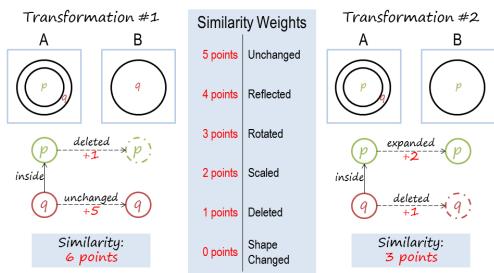


Figure 86: Choosing Matches by Weights

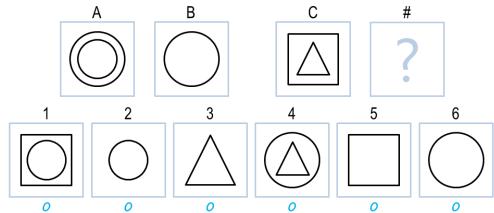


Figure 87: Choosing Matches by Weights

So let us look at the semantic network representation of the relationship between A and B. In one view of the transformation from A to B, we can think of q, the outer circle, as remaining unchanged, and p the inner circle, as getting deleted. Let's look at another view of the transformation from A to B. In this view, we can think of p as getting expanded and q, the outer circle, as getting deleted. Both of these views are valid views. If both of these views are valid, then how would anyone decide? How would an AI agent decide which view to select? Let us suppose that

the AI agent had a metric by which it could decide upon the ease of transformation from A to B. Let us suppose that, that metric assigned different weights to different kind of transformations. You will notice that these transformations like scaling, rotation, reflection make for a fine transformations. In this scale, a larger value like 5 points, means more ease of transformation and greater similarity. A lower value means less ease of transformation and more difficult transformation and less similarity. Given the scale, let us calculate the weight of transformations for both transformation #1, and transformation #2. In transformation #1, you can see that p is getting deleted, which we gave a weight of 1. And q remains unchanged, which we gave a weight of 5. So the total weight here is 6. In case of transformation #2, the weight of p being expanded, we said will be 2, scaled. And, q getting deleted is 1, so the total weight is 3. If you prefer the first transformation over the second transformation, then we can see why someone will answer the square is the correct answer, and not the triangle. Let us return to this exercise. And now we can see why both 3 and 5 are legitimate answers. We can also see why an AI agent may prefer 5, given the similarity metric that we talked about in the last shot.

27 - Discussion Choosing a Match by Weight

[Click here to watch the video](#)

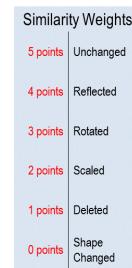


Figure 88: Discussion Choosing a Match by Weight

In order to make sure that they're getting the point here let us revisit an exercise we had come across earlier. This is a truncated version of the exercise. That one had six choices, here there

are only four choices. The similarity weights are shown here. David, which one do you think is the right choice for this problem? So it seems to me that given these four choices, the right choice is going to be number 2. What does everyone think about David's answer? Did David give the right answer with 2?

28 - Discussion Choosing a Match by Weight

[Click here to watch the video](#)

What is everyone think? Is 2 the right answer here? Well, lets look at the choices. First note, that both 2 and 4 are legitimate answers. 2 is legitimate because we can think of the transformation from A to B as a reflection around the vertical axis. And so if we think of the transformation from C to D, again as a reflection of the vertical axis, we'll get 2. 4 is also a correct answer, because we can think of the transformation from A to B as a rotation of 180 degrees, and if we rotate C by 180 degrees we'll get 4. However, if we look at our weights again, we gave reflection a higher weight than we gave rotation. And therefore, David is right, 2 indeed is the correct answer.

29 - Connections

[Click here to watch the video](#)

Before we end this lesson, I want to draw several connections. The first has to do with memory. We have often said that memory is an integral part of the cognitive systems architecture. One can imagine that A and B are stored in a memory. Then C and 1, and C and 2, and C and 3, and so on, are probes into the memory. And the question would then become, which one of these probes is most similar to what's stored in memory? We may decide on that answer based on some similarity metric. In fact we'll revisit this exactly the same issue when we talk about case-based reasoning later on in this class. Another connection we can draw here has to do with reasoning. When we are talking about the transformation from A to B and then the transformation to from C to 1 of these choices, one question that arose was should we make the connection

between the outer circle here and B? Or the inner circle and A and B. This is a correspondence problem. The correspondence problem is: given two situations, what object in one situation corresponds to what object in another situation? We will come across this problem again when we discuss analogical reasoning a little bit later. The third connection has to do with cognition, of knowledge based AI as a whole. Notice that instead of just talking about properties of objects, like this is a circle and the size of the circle, our emphasis here has been on the relationships between the objects. The fact that this is inside the outer circle, or the fact that the outer circle remains the same here, the inner circle disappears. In knowledge-based AI and in cognition in general, the focus is always on relationships, not just on objects and the features of those objects.

30 - Assignment Semantic Nets

[Click here to watch the video](#)

Assignment

How would you use semantic networks to design an agent that can answer Raven's Progressive Matrices?

Figure 89: Assignment Semantic Nets

The first assignment you can choose to do for this course is to talk about how these semantic networks can be used to represent Raven's Progressive Matrices. We saw a few different problems in the first lesson. So take a look at how the semantic networks we've talked about today can be used to represent some of those other problems, and write your own kind of representation scheme. In addition to writing a representation scheme, also talk about how that representation scheme actually enables problem solving. Remember what Ashok mentioned about the different qualities of a good knowledge representation that is complete. It captures information at the right level of abstraction, and it enables

problem solving. So write how your representation can enable problem solving of two by one, two by two, and three by three problems. You don't need to use the exact same representation scheme that we use, and in fact you can and should use your own. Also remember that your representation should not capture any details about what the actual answer to the problem is, but rather it should only capture what's actually in the figures in the particular problem.

31 - Wrap Up

[Click here to watch the video](#)

To recap...

- Representations
- Semantic networks
- Represent & Reason
- Weights with Represent & Reason

Figure 90: Wrap Up

So let's recap what we've talked about today. We started off today by talking about one of the most important concepts in all of knowledge based AI, which are knowledge based representations. As we go forward in this class, we'll see knowledge representations are really at the heart of nearly everything we'll do. We then talked about semantic networks, which are a good particular kind of [knowledge] representation and we used those to talk about the different criteria for a good knowledge representation. What do good knowledge representations enable us to do and what do they help us avoid? We then talked about kind of an abstract class of problem solving methods called Represent and Reason. Represent and reason really lies under all of knowledge based AI and it's a way of representing knowledge and then reasoning over it. We then talked a little bit about augmenting that with weight, which allows us to come to more nuanced and specific conclusions. In the next couple weeks, we are going to use these semantic networks to talk about a few different problem

solving methods. Next time, we'll talk about generating tests and then we'll move on to a couple slightly different ones called Means and Analysis and Proper Reduction.

32 - The Cognitive Connection

[Click here to watch the video](#)

How is semantic networks connected with human cognition? Well we can make at least two connections immediately. First, semantic networks are kind of knowledge representation. We saw how knowledge is presented as a semantic network. If the results of [power of] representation, then you can use the knowledge presentation to address the problem. We can now say similarly for human mind that human mind represents problems. It represents knowledge. Then it uses that knowledge to address the problem. So, representation then becomes the key. Second, and most specifically, semantic networks are related to spreading activation networks, which is a very popular theory of human memory. Let me give you an example. Supposing I told you a story consisting of just two sentences. John wanted to become rich. He got a gun. And notice that I did not tell you the entire story, but I'm sure you all made up a story based on what I told you. John wanted to become rich. He decided to rob a bank. He got a gun in order to rob the bank. But how did you end this story? How did it draw the inferences about robbing a bank which I did not tell you anything about? Imagine if you have a semantic network that consisted of a large number of nodes. So when I gave you the first sentence, John wanted to become rich, the nodes corresponding to John and wanted and become and rich, got activated, and the activation started spreading from those nodes. And when I said John, he got a gun, then the gun node also got activated and that activation also started spreading. As this activation spread, it merged. And a path they could walk on. And all the nodes on that pathway now become part of the story, and if you happen to have nodes like, rob a bank along the pathway, now you have understanding of story.

33 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

34 - Final Quiz

[Click here to watch the video](#)

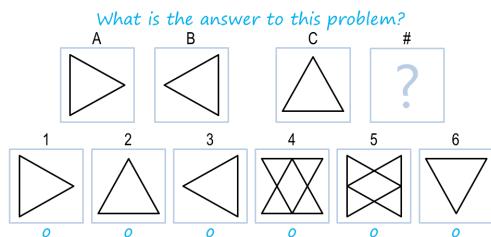


Figure 91: Final Quiz

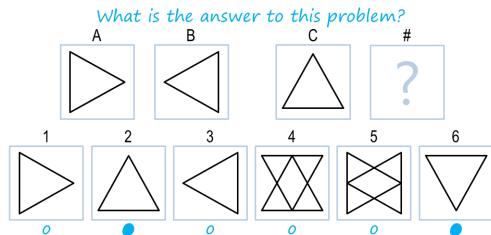


Figure 92: Final Quiz

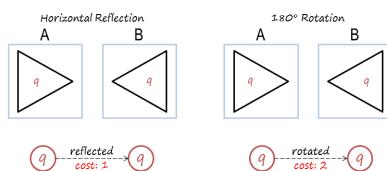
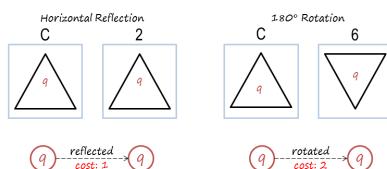


Figure 93: Final Quiz



What would the answer be? ● 2 ○ 6

Figure 94: Final Quiz

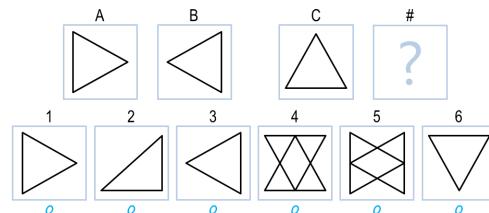


Figure 95: Final Quiz

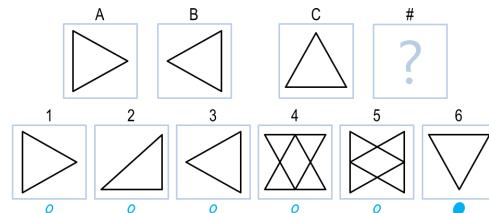


Figure 96: Final Quiz

Great. Thank you so much for your feedback.

Summary

This lesson cover the following topics:

1. Knowledge representation and Reasoning using that representation is the key to problem-solving.
2. Semantic Networks are one of the many ways for knowledge representation.
3. Pathways along spreading activation networks could potentially help with memorizing and recalling solutions instead of solving them every time for new recurring problems.

References

1. Winston P., Artificial Intelligence, Chapter 2, Pages 15-33.

Optional Reading:

1. Winston Chapter 2, pp. 16-32; [Click here](#)

Exercises

None.

Lesson 04 - Generate & Test



It is evidently necessary to generate and test candidates for solutions in some systematic manner.

– Niklaus Wirth: Swiss computer scientist, creator of the Pascal language.

01 - Preview

[Click here to watch the video](#)

Fundamentals

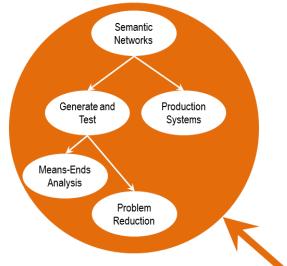


Figure 97: Preview

Given a problem, generate potential solutions to it, and then test the solutions for the efficiency for addressing the problem. If you had complete knowledge of the world, complete and correct knowledge of the world. If you had infinite computational resources. And if you had a method of reasoning that was guaranteed to be correct, then you can generate the correct, the optimal solution. You don't need to test it. In general however, you do not have complete or correct knowledge of the world. You do not have infinite computational resources, and not all methods are guaranteed to be correct. In that case, you can generate potential solutions. Plausible solutions, and then test them. Estimates that will generate, and test. We will use this method in conjunction with semantic networks, or the prisoners and guards problem that we discussed last time.

Lesson Preview

- Generate and test method
- Smart testers
- Smart generators
- Generate and test for Raven's Progressive Matrices

02 - Guards and Prisoners

[Click here to watch the video](#)

Figure 98: Preview

Today, we'll talk about a very general method called generate and test. We have looked at knowledge representations like semantic networks. We'll shift our attention now to problem solving methods. Generate and test is a problem solving method. This is another item in our fundamental topics part of the course. The generate and test method in a way, is very simple.

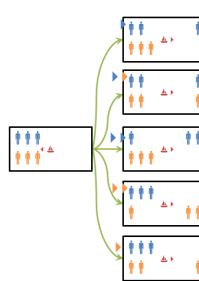


Figure 99: Guards and Prisoners

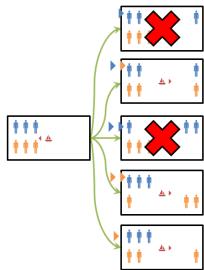


Figure 100: Guards and Prisoners

Knowledge-based AI is a collection of three things. Knowledge representations, problem solving techniques and architectures. We have already looked at one knowledge representation, semantic networks. We have not so far looked at problem solving methods or architectures. Today, I'd like to start by talking about the problem solving method. Let us illustrate the problem solving method of generate and test with the same examples that we have discussed earlier. When we were discussing this example in the case of semantic networks, we simply came up with various states and pruned some of them without saying about how an AI agent would know what states to prune. So imagine that we have a generator that takes the initial state and from that initial or current state, generates all the possible successor states. For now, imagine it's not a very smart generator, it's a dumb generator. So it generates all the possible states. So the generator test method not only has a generator but also has a tester. The tester looks at all the possible states the generator has generated and removes some of them. For now, let's also assume that the tester is is dumb as well. And so the tester is removes only those states that are clearly illegal based on the specific of the problem. Namely, that one cannot have more prisoners than guards on either back. So the first and the third states are removed by the tester.

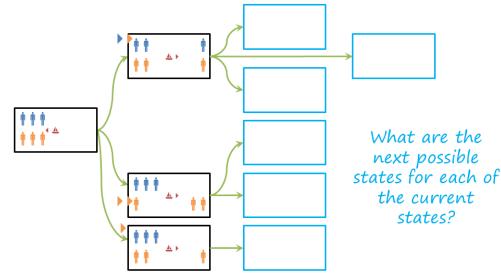


Figure 101: Exercise Generate and Test I

Let us continue with this exercise one step further. So now we have three successor states to the initial state. Given these three successor states, what states might the dumb generator generate next?

04 - Exercise Generate and Test I

[Click here to watch the video](#)

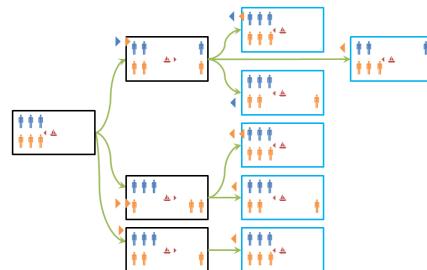


Figure 102: Exercise Generate and Test I

So from the top state we have three possible next states. We can move both of them, we can move just the prisoner, or we can move just the guard. From this one we can either move one prisoner or two prisoners, and from this one all we can really do is move the prisoner back over to the left. Remember that David is not generating these successive states. David is saying that the DOM generator will generate the successive states.

03 - Exercise Generate and Test I

[Click here to watch the video](#)

05 - Exercise Generate and Test II

[Click here to watch the video](#)

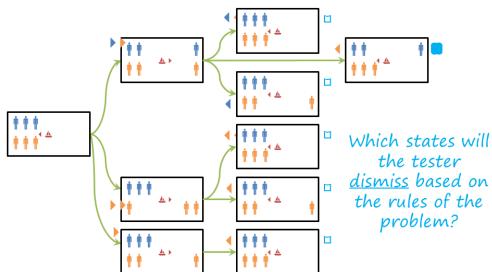


Figure 103: Exercise Generate and Test II

So now that we have all of these states that the generator has generated, given that we have a dumb tester what states will the dump tester dismiss?

06 - Exercise Generate and Test II

[Click here to watch the video](#)

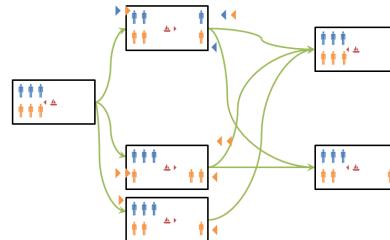


Figure 106: Exercise Generate and Test II

So the only one of these six states that disobeys our one rule against having more prisoners than guards on either shore, is this state over here. So, that's the only state that's going to get thrown out. These five states are all legal according to our dumb testers understanding of the problem. So after we dismiss that state, though. We'll notice that we only have two unique states, we have everyone on the left coast and one prisoner on the right coast. So like we did earlier, we can collapse these two down into only these two states. It won't matter how we got there, once we're there.

07 - Dumb Generators, Dumb Testers

[Click here to watch the video](#)

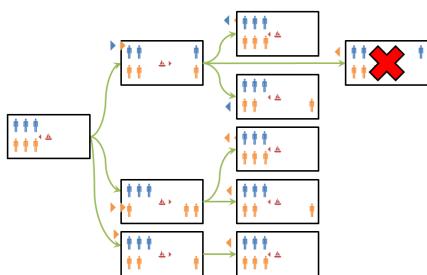


Figure 104: Exercise Generate and Test II

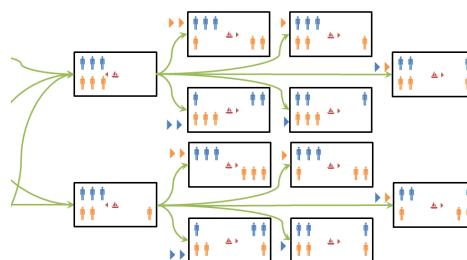


Figure 107: Dumb Generators, Dumb Testers

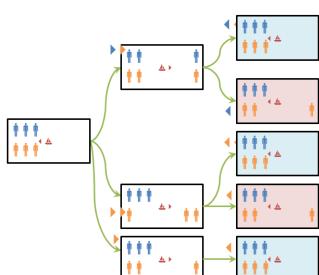


Figure 105: Exercise Generate and Test II

Now we can continue to apply this method of generate and test iteratively. So we can apply it on this state and that state and see what successor states we get. If we do so, then we get a very large number of successor states. This is a problem of call many total explosion. While one was tasked with a small number of states, but the number of successor states keeps on increasing very rapidly. Now, the reason it is occurring here and it did not occur when we are

talk, dealing with semantic networks is because here we have states like this one which have three guards and three prisoners on the same side of the bank, exactly the same state that was the initial state to begin with. This is because we have a dumb generator and a dumb tester. So this state never got pruned away, although this particular state is identical to the initial state that we started from. This method of generating test, even with a dumb generator and a dumb tester, if applied iteratively could finally lead to the goal state. In which case, we will have a path from the initial state all the way to the goal state, but this will be computationally very inefficient. This is because we have a dumb generator and a dumb tester. So the question now becomes, can we make a smarter generator and a smarter tester? Before we make a smarter generator and a smarter tester, we should note that generate and test is a very powerful problem solving method.

08 - Smart Testers

[Click here to watch the video](#)

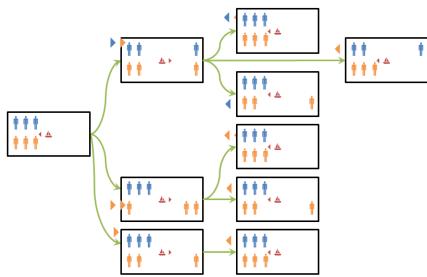


Figure 108: Smart Testers

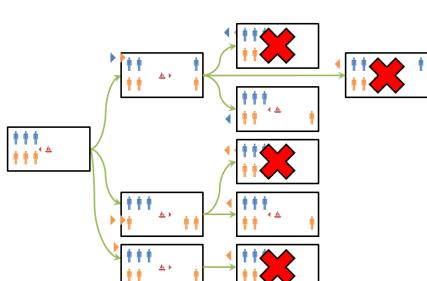


Figure 109: Smart Testers

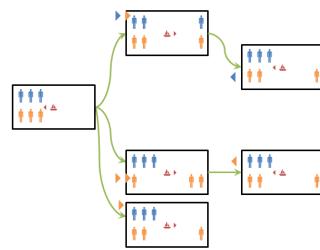


Figure 110: Smart Testers

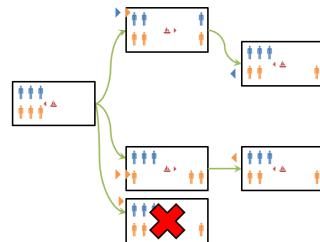


Figure 111: Smart Testers

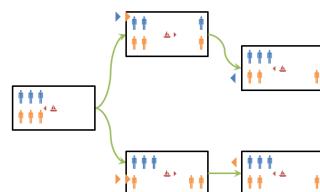


Figure 112: Smart Testers

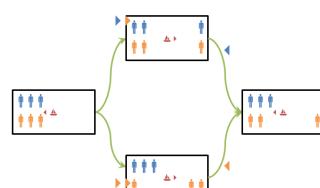


Figure 113: Smart Testers

So suppose that we have a smarter tester, a tester which can detect when any state is identical to a previously visited state. In that case

the tester may decide that this, this, and this state are identical to the initial state and therefore dismiss them. The tester also dismisses this state, as usual, because of the problem specification that one cannot have more prisoners than guards on any one bank. This leaves the following state of affairs. Note also that this particular state has no successor states, all successor states of this have been ruled out. Therefore this particular part clearly is not a good path to get to the gold state. If we notice also, that these two states are identical, then we can merge them. If we do so, then we get exactly the same kind of configuration of states that we had when we were dealing with the semantic network in the previous lesson. There is something to note here. We had this semantic network in the last lesson, but the knowledge representation of semantics network, while very useful, by itself and of itself doesn't solve any problems. You need a problem solving method that uses knowledge afforded by the knowledge representation to actually do the problem solving. Generating test is one of those problem solving methods. In general, when we do problem solving or reasoning, then there is a coupling between a knowledge representation and a problem solving method, like semantic networks and generating test. What we did so far had a dumb generator, but we made the testers smarter. The testers started looking for what states had been repeated. Alternatively we can shift the balance of responsibility between them and make the generator smarter. Let's see how that might happen.

09 - Smart Generators

[Click here to watch the video](#)

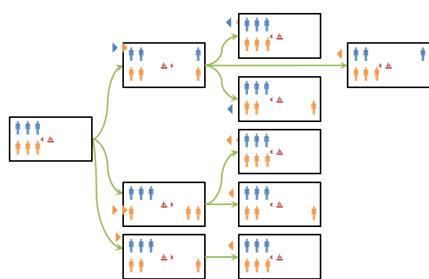


Figure 114: Smart Generators

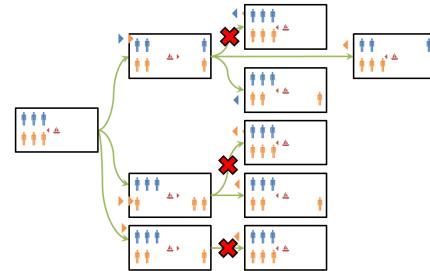


Figure 115: Smart Generators

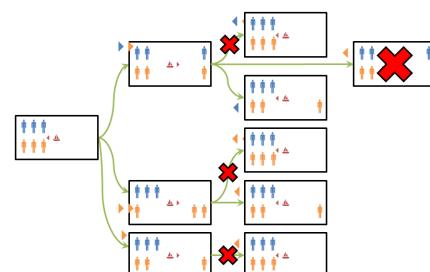


Figure 116: Smart Generators

Instead of the generator generating all the successive states and then a tester finding out that this state, this state and this state are identical to the initial state. One could make the generator itself smarter and say that a generator will not even generate these three states, but it will know that it should not generate states that are already up here. This means that we can either provide the generator with some additional abilities or the tester with some additional abilities or both. If the generator was smarter, then it would not even generate these three states because they are nonproductive. I would exclude maybe the tester, the determinant of this state is illegal and therefore dismisses it. We could even go one step further and make the generator even smarter, so the generator will not generate this particular state. And thus, the balance within the generator and the tester can shift depending on where we try to put knowledge. For this problem, for this relatively simple and small problem, the balance will responsibility between the generator and test might look like a tree relationship. But imagine a problem in if there are a million such states. Then whether we have generated very smart or the tests are very smart

or both can become an important issue. Despite that, genetic testing factors are a very popular method used in some schools of AI. Genetic algorithms, for instance, can be viewed as genetic generate and test. Given a number of states, they try to find out all the potential successive states that are possible, given some simple rules of recombination. And then of a fitness function that acts as a tester. Genetic algorithms, therefore, are an effective method for a very large number of problems. They're also a very inefficient method because neither the generator nor the testing generator algorithms are especially smart.

10 - Discussion Smart Generators and Testers

[Click here to watch the video](#)

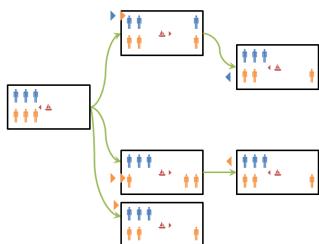


Figure 117: Discussion Smart Generators and Testers

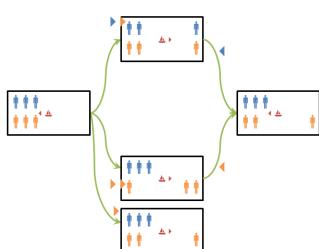
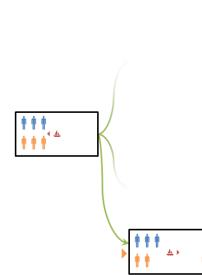


Figure 118: Discussion Smart Generators and Testers



Is David right?

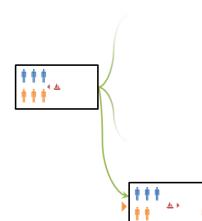
- Yes, this is the only way to throw out that state.
- Yes, but there is a rule to prevent the state from being generated in the first place.
- No, because the rule should only apply to the right coast because we are trying to move everyone to the right.
- No, because that rule would throw out some valid states as well.

Figure 119: Discussion Smart Generators and Testers

Recall that we had this state as a kind of hanging state. It had no successes state. And we said that, well this state can be dismissed, but here is a question. Who will dismiss it, the generator or the tester? What do you think, David? So I think the tester would do it. We could have the tester basically say any state where there is only one person on the side of the river that has the boat can be pruned off, because the only way to get into that state is to send that person over. And the only way to get out of that state is to undo the previous move. So my rule would be for the tester, any time only one person is on the side of the river with the boat, throw out that state. What does everyone else think? Is David right about this?

11 - Discussion Smart Generators and Testers

[Click here to watch the video](#)



Is David right?

- Yes, this is the only way to throw out that state.
- Yes, but there is a rule to prevent the state from being generated in the first place.
- No, because the rule should only apply to the right coast because we are trying to move everyone to the right.
- No, because that rule would throw out some valid states as well.

Figure 120: Discussion Smart Generators and Testers

David is right. The tester could, in fact, test and dismiss the state using the kind of rule David was talking about. But, David, do you think there is another way that this could be done?

So, I guess, actually, the generator could do it as well. So, in this case, we could also have the generator say, I don't need to ever generate a state that involves sending one person over to an empty coast. That's always going to result in a state that the tester would have thrown out. And it's not going to result in any states that the tester wouldn't have thrown out. So, generator could do it, too. That sounds like a good answer to me. So once again, we are back to the issue of where do we draw the balance of responsibility between the generator and the tester? The important thing to note from here, however, is that generate and test, when endowed with the right kind of knowledge, can be a very powerful method.

12 - Generate Test for Ravens Problems

[Click here to watch the video](#)

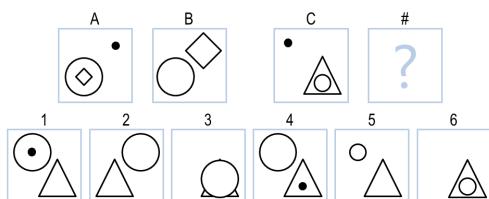


Figure 121: Generate Test for Ravens Problems

Figure 122: Generate Test for Ravens Problems

Let us return to our problem from the intelligence test to see how generate and test might apply as a problem solving method. Again, here is a problem that we encountered earlier. Notice that this is a more complicated problem than the guards and prisoners problem. Here is why. In

case of the guards and prisoner problem, each transformation from one state to another, was a discrete transformation. One could take a certain number of guards to the other side. One could take a certain number of prisoners to the other side, or one could take a certain number of guards and prisoners to the other side. In this case, if I look at the approximation between A and B, and I notice that the diamond inside the circle is now outside the circle and is larger. Now suppose I were to try the same transformation from C to D. So I can look at the circle inside the triangle, put it outside, and also make it larger. I notice that when I put it outside, I can put it outside right next to the triangle, a little bit farther, a little bit farther, a little bit farther away. I can make it the same size, or a little larger, or a lot larger. Increase its size by 50

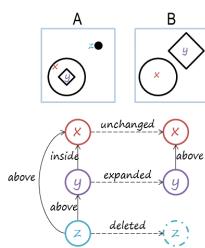
13 - Semantic Networks for Generate and Test

[Click here to watch the video](#)

Assignment

How would you use the generate and test method to design an agent that could answer Raven's Progressive Matrices?

Figure 123: Semantic Networks for Generate and Test



This is where the knowledge representation helps a lot. The semantic network knowledge representation provides a level of abstraction at which the problem gets represented and analyzed. So, although this particular diamond y could have been displaced here or a little bit further, it could have been of this size, maybe a little smaller, a little bit larger. The semantic network really doesn't care about it. With the level of extraction which a semantic network is dealing, y gets expanded, and that is all that matters. An important point to note here is that any knowledge representation picks a level of extraction at which it represents the world.

There's a lot of power in it because that knowledge representation ignores things that are at a low level of detail. And therefore the problem-solving method doesn't have to worry about those things. So it is not the knowledge representation alone that solves the problem, or the problem solving method that solves the problem. It is the knowledge representation and the problem solving method coupled together that solve the problem, that provide the reasoning.

14 - Generate Test for Ravens Problems II
[Click here to watch the video](#)

Good Generators

- Complete
- Efficient
- Smart

To recap...

- Generate and test
- Smart testers
- Smart generators
- Generate and test in an unconstrained domain

Figure 124: Generate Test for Ravens Problems II

which one most closely matched what I generated. If I wanted to make my tester and generator even smarter, I might say that in order to be the correct answer, it has to meet the generated answer with a certain level of confidence. And if it doesn't meet that level of confidence, it should go back and see if there's a different transformation we could have transferred. That would take care of the problem earlier where either the middle shape disappeared or the outer shape disappeared. That's a good answer, David. It is another way of solving this problem. It is another way of solving this problem using test and semantic networks. One could take one, put it under D. Generate the transformation from C to D and then test it against the transformation from A to B. One could do the same thing with 2, put 2 here into D, directly transformation tested against the transformation A to B. One could do this for all six choices and then find out, which one of these transformations is closest with the transformation from A to B. Thus, in this problem, one can use generate and test methods in two very different ways, all of the knowledge representation ribbon is the same. So knowledge representation captures some knowledge about the world at a level of abstraction. It is coupled with problem solving methods, but more than one problem solving method, more than one variation of a problem solving method might be applicable using technology representation.

15 - Assignment Generate Test
[Click here to watch the video](#)

Figure 125: Generate Test for Ravens Problems II

So let's assume that we're using semantic network as a representation for this particular class of problem. Given that, how would you apply generate and matter to this problem, David? So it sounds like would I would do is I would use the transformation between A and B, transfer that transformation to C and use it to generate my answer for D. I then take my answer for D and compare it against 1, 2, 3, 4, 5 and 6 and see

So how would you use generate and test to actually solve Raven's Progressive Matrices? We've talked about this a little bit already, but take it a little bit further and talk about how you would actually implement the problem solving approach you've seen today. We talked about a couple different ways of going about it. We talked about generating multiple answers and testing them against the answer options. Or generating one answer and testing it more intelligently against the different options available to you. So talk about which one you would do and how you would actually implement it. In doing

so, make sure to think of three by three problems as well. With more transformations and more figures going on, it can be a lot more difficult to figure out what to generate, and the problem space can explode very quickly. Also make sure to think about how you're actually going to infer the mapping between different figures in the problem. How do you know which shape in one frame maps up to a different shape in another frame? And then talk about how you would use that information to generate what you think the answer is.

16 - Wrap Up

[Click here to watch the video](#)

Let's wrap up our topic for today. So today, we've talked about generate and test, which is a very general purpose problem solving method. As Ashok mentioned earlier in our lesson, we see generate and test every day in our regular lives and it's something in which we engage very naturally. We talked about strong generators and strong testers and how we can build intelligence into one side or the other in order to make the problem solving process easier and more efficient. We also talked about how generating tests is more difficult and unconstrained domains and how our generator and tester need to be equipped with special kinds of knowledge in order to make this problem solvable. Next, we're going to be looking at two different problem solving methods that build on what we've seen today with generate and test. Means and analysis and problem reduction. Like generate and test, these are both very general purpose, but they're going to do things a little bit differently and make certain problems easier.

17 - The Cognitive Connection

[Click here to watch the video](#)

Let us examine the relationship between the method of generate and test, and human cognition. Humans use generate and test as the problem-solving method all the time. This is because we do not have complete or correct knowledge of the world. We do not have infinite computational resources. And we also do not always

have recourse to a method of reasoning that is guaranteed to be correct. When you do not have these things, then you use your own test method. You come up with particular solutions to a problem, you test the solutions out. Beyond human cognition, I'm sure you've come across the notion of genetic algorithms. Genetic algorithms are inspired by the processes of biological evolution. Through operations like crossover and mutation, one can generate solutions that can then be tested against some fitness function. Genetic algorithm are a good example of the genetical test method. First, genetic solutions, then test them out. So this method of generating test is connected not only with human cognition, but dependently, also with biological evolution. It's all over the place.

18 - Final Quiz

[Click here to watch the video](#)

Once again, will you please complete the quiz at the end of this lesson? What did you learn in this lesson?

19 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

This lesson cover the following topics:

1. Generate and Test is a very commonly used problem-solving method used by humans and in nature by biological evolution (similar to Genetic algorithms).
2. We need both knowledge representation and problem-solving methods together to provide reasoning to solve problems.
3. Smart generators and smart testers help prune multitude number of states that are possible due to combinatorial explosion of successor states, thereby helping solve intractable problems efficiently using limited computational resources and limited knowledge of the world as compared to dumb generators and dumb testers.

References

1. Winston P., Artificial Intelligence, Chapter 3.

Optional Reading:

1. Winston Chapter 3, pp. 47-50; [Click here](#)

Exercises

Exercise:

You are working on your computer in your apartment when you get attacked by an army of ants. You run to the nearest supermarket and go aisle to aisle looking for a useful weapon – foods, beverages, cosmetics, medicines, alcohol, ice cream, anything. You are using the strategy of generate and test of course, but does your specific strategy the three properties of a good generator?

Lesson 05 - Means-Ends Analysis



In the final analysis, means and ends must cohere because the ends is preexistent in the means, and, ultimately, destructive means cannot bring about constructive ends.

– Martin Luther King: American leader in the African-American Civil Rights Movement.

01 - Preview

[Click here to watch the video](#)

Fundamentals

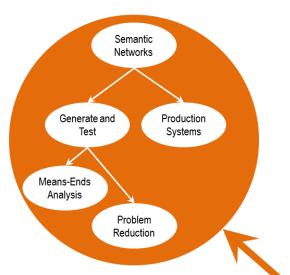


Figure 126: Preview

well-formed problems. Not all problems are well-formed. But some problems are. And then these methods are very useful. These three methods, generate and test, means-ends analysis, form reduction, together with semantic networks as a knowledge representation, form the basic unit of all fundamental topics in this course. We'll begin with the notion of state spaces. Then talk about means-end analysis. Then we'll illustrate means-end analysis as a matter for solving problems and then we'll move onto the method of problem reduction.

Lesson Preview

- State spaces
- Means-ends analysis
- Problem solving with means-ends analysis
- Problem reduction

Figure 127: Preview

Today we will discuss two other very general AR methods of problem solving called means end analysis and problem reduction. Like generate and test these two methods, means analysis and problem reduction, are really useful for very

02 - Exercise The Block Problem

[Click here to watch the video](#)

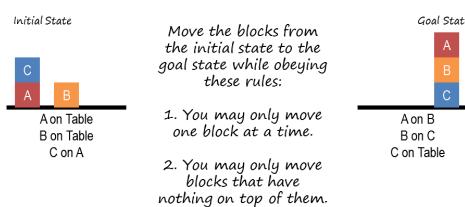


Figure 128: Exercise The Block Problem

LESSON 05 - MEANS-ENDS ANALYSIS

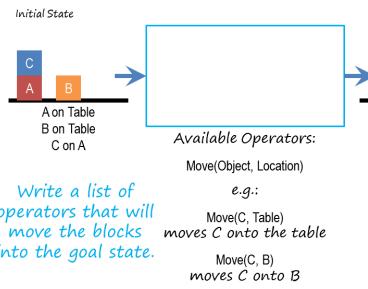


Figure 129: Exercise The Block Problem

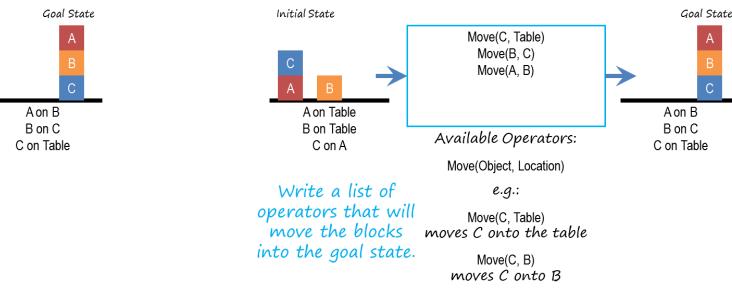


Figure 130: Exercise The Block Problem

To understand a method of means and analysis. Let us look at this blocks word problem. This is a very famous problem in AI. It has occurred again and again. And almost every textbook in AI has this problem. You're given a table on which there are three blocks. And A is on table, B is on table, and C is on A. This is the initial state. And you want to move these blocks, to the gold state. On this configuration, so that C is on table, B is on C and A is on B. The problem looks very simple listen, doesn't it? Let's introduce a couple of constraints. You may move only one block at a time, so you can't pick both A and B together. And second, you may only move a block that has nothing on top of it. So, you cannot move block A in this configuration, because it has C on top of it. Let us also suppose that we're given some operators in this world. These operators essentially move some object to some location. For example, we could move C to the table, or C onto B, or C onto A. Not all the operators may be applicable in the current state. C is already on A, but in principle, all these, all of these operators are available. Given these operators, and this initial state and this goal state, write a sequence of operations that will move the blocks from the initial state to the goal state.

03 - Exercise The Block Problem
[Click here to watch the video](#)

That's a good answer, David, that's a correct answer. Now the question becomes how can we make in AI agent that will come up with the similar sequence of operations? In particular, how does the matter of means-end analysis work on this problem and come up with a particular sequence of operations?

04 - State Spaces
[Click here to watch the video](#)

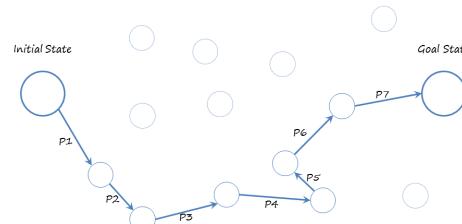


Figure 131: State Spaces

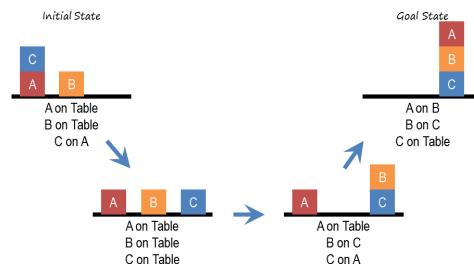


Figure 132: State Spaces

So, we can imagine problem solving as occurring in a state space. Here is the initial state, here is the goal state. And the state space consists of all of the states that could be potentially

produced from the initial state by iterative application of the various operators in this micro world. I want to come up with a path in the state space, takes me from initial state to the goal state. There is one path, this is not the only path, but this is one path to go from the initial state to the goal state. The question then becomes, how might an AI agent derive this path that may take it from the initial state to the goal state. Let us see how this notion of path finding applies to our blocks world problem. From the initial state, here it is one path of going to the goal state. First, we put C on the table. Then we put B on top C. And then we put A on top of B. Which is exactly the answer that David had given. This is one sequence, one path from the initial state to the goal state. The question then becomes, how does AI method know what operation to select in a given state? Consider this state, for example. There are several operations possible here. One could put C on top of B or B on top of A. How does the AI agent know which operation to select at this particular state?

05 - Differences in State Spaces

[Click here to watch the video](#)

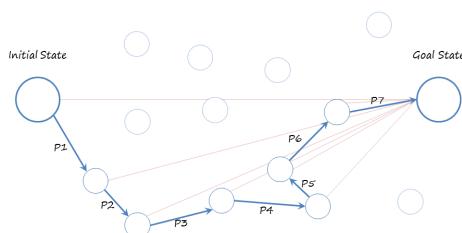


Figure 133: Differences in State Spaces

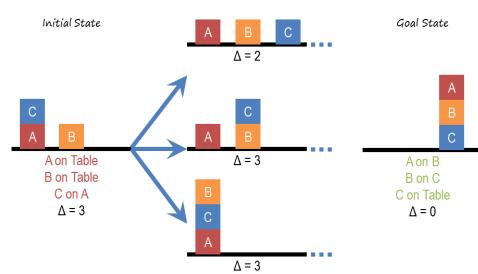


Figure 134: Differences in State Spaces

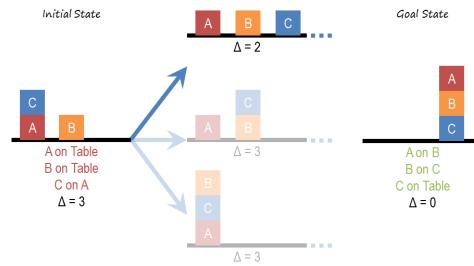


Figure 135: Differences in State Spaces

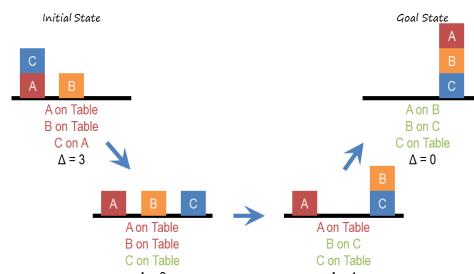


Figure 136: Differences in State Spaces

One way of thinking about this is to talk in terms of differences. This chart illustrates the differences between different states and the goal state. So, for example, if the current state was this one then this red line illustrates the difference from the goal state. So we should pick an operator that will help reduce the difference between the current state and the goal state. So the reduction between the difference with the current state and the goal state is the end. The application of the operator is the means. That's why it's called the means-ends analysis. At any given state, I'm going to pick an operator that will help you deduce the difference between the current state and the goal state. Note in a way this problem is similar to the problem of part finding in robotics, where we have to design a robot that could go from one point to another point in some navigation space. From my office to your office, for example, if all our offices were in the same building. There too we would use the notion of distances between offices. Here we using the notion of distance in a metaphorical sense, in a figurative sense, not in a physical sense. So I'll sometimes use the word difference instead of distance but it's the same idea. We

are trying to deduce the distance or the difference but in an abstract space. So going back to an example of going from this initial state to this goal state. I can look at initial state and see that there are three differences between the initial state and the goal state. First, A is on table here, but A should be on B. B is on table here, but B should be on C. And third, C is on top of A here, the C should be on top, on table there. So three differences. Here the number of operations are available to us. Nine operations in particular. Let us do a means-end analysis. We can apply an operator that would put C on table. In which case the difference between the new state and the goal state will be two. We could apply an operator that will put C on top of B, in that case the difference between the current state and the goal state will still be three. Or we can apply the operator putting B on top of C, in which case the distance between the current state and the goal state will be 2. Notice that the notion of reducing differences now leads to two possible choices. One could go with this state or with this one. Means-end analysis by itself does not help an AI agent decide between this course of action and that course of action. This is something that we will return to, both a little bit later in this lesson and even much more in detail when we come to planning in this course. For now, let us resume that we choose the top course of action just like they had done already there. So this chart illustrates the path taken from the initial state to the goal state. And the important thing to notice here is that with each different move the distance between the current state and the goal state is decreasing, from three to two to one to zero. This is why means-end analysis comes up with this path because at each time it reduces a difference

Means-Ends Analysis

For each operator that can be applied:

- Apply the operator to the current state
- Calculate difference between new state and goal state

Prefer state that minimizes distance between new state and goal state

Figure 137: Process of Means End Analysis

We can summarize the means-ends analysis method like this. Compare the current state and the goal state. Find the differences between them. For each difference, look at what operators might be applicable. Select that operator that gets you closest to the goal state from the current state. We did this for the blocks and worlds problem. We also did this with regards to the business problem. But throughout those states in regards to business problem, which we're not getting us close to the goal state. This is the means-ends analysis method in summary.

07 - Exercise Block Problem I

[Click here to watch the video](#)



Figure 138: Exercise Block Problem I

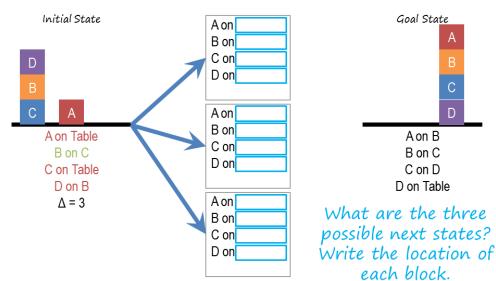


Figure 139: Exercise Block Problem I

06 - Process of Means End Analysis

[Click here to watch the video](#)

To understand more deeply the properties of means and analysis, let us look at another, slightly more complicated example. In this example, there are four blocks instead of the three in the previous example. A, B, C, D. In the initial state, the blocks are arranged as shown here. The goal state is shown here on the right. The four blocks are arranged in a particular order. Now if you compare the configuration of blocks on the left with the configuration of blocks on the right, in the goal state, you can see there are three differences. First, A is on Table, where A is on B here. B is on C. That's not a difference. C is on Table. C is on D here, D's on B, D's on Table here. So there are three differences. So, this is a heuristic measure of the difference between the initial state and the goal state. Once again, we'll assume that the AI agent can move only one block at a time. Given the specification of the problem, what states are possible from the initial state? Please write down your answers in these boxes.

08 - Exercise Block Problem I

[Click here to watch the video](#)

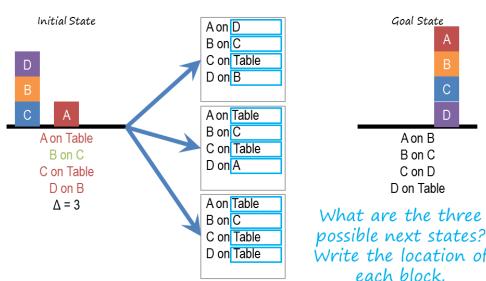


Figure 140: Exercise Block Problem I

That's good David.

09 - Exercise Block Problem II

[Click here to watch the video](#)

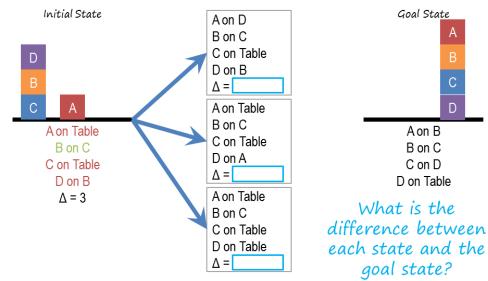


Figure 141: Exercise Block Problem II

Okay now for each of these states that is possible from the initial state what are the differences as compared to the goal state? Please write down your answers in these boxes.

10 - Exercise Block Problem II

[Click here to watch the video](#)

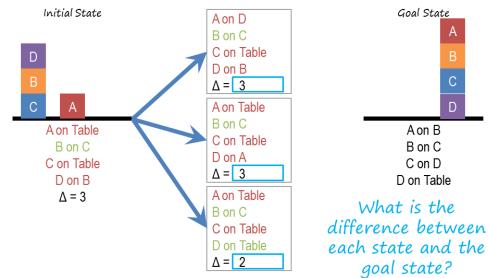


Figure 142: Exercise Block Problem II

What answers did you come up with David? So for the first one, if we put A up on D, we haven't accomplished any of the goals that we hadn't already accomplished. So our difference is still 3. Similarly, for putting D on A, our difference is also still 3, because then we haven't accomplished anything new. However, if we put D down on the table, we've accomplished one of the goals we hadn't accomplished before. So there, our difference is 2. Good, David. So, in each state, David is comparing the state with the goal state, and finding differences between them.

11 - Exercise Block Problem III

[Click here to watch the video](#)

LESSON 05 - MEANS-ENDS ANALYSIS

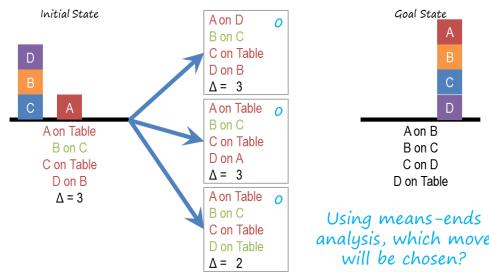


Figure 143: Exercise Block Problem III

Given these three choices which operation would means-end analysis choose?

12 - Exercise Block Problem III

[Click here to watch the video](#)

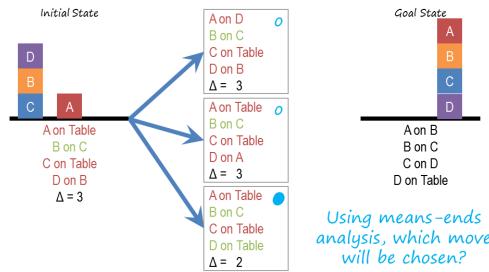


Figure 144: Exercise Block Problem III

What was your answer David? So means-ends analysis always chooses the state that most reduces the distance to the goal state. In this case, that would be the third option, because it reduces the distance down to two. That's correct David.

13 - Exercise Block Problem IV

[Click here to watch the video](#)

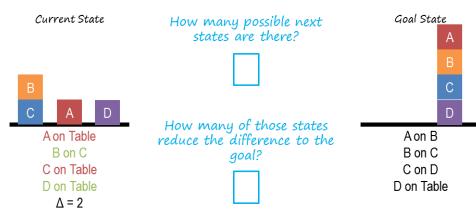


Figure 145: Exercise Block Problem IV

Given this current state, we can apply means ends analysis veritably. Now, if we apply means on some of those to this particular state, the number of choices here is very large, so I will not go through all of them here. But I'd like you to write down the number of possible next states. As well as, how many of those states reduce the difference to the goal? Which is given here.

14 - Exercise Block Problem IV

[Click here to watch the video](#)

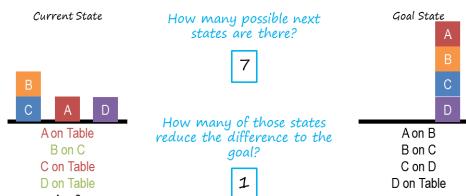


Figure 146: Exercise Block Problem IV

What answers did you come up with, David? So I counted out seven possible next states. We can put B on A, B on D, B on the table, A on B, A on D or D on B or D on A. So that's seven total operations. Of those, only one of them accomplishes a goal we hadn't already accomplished, and that's putting A up on B. That's good, David.

15 - Exercise Block Problem V

[Click here to watch the video](#)

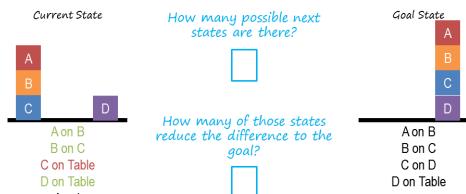


Figure 147: Exercise Block Problem V

So, the operation of putting A on B will bring us to this state. Given this state, we can have, again, apply a means of analysis. Again, I'm not

sure that all these states here, but I'd like you to find out how many possible states are there and how many of those states reduce the difference to the goal described.

16 - Exercise Block Problem V

[Click here to watch the video](#)

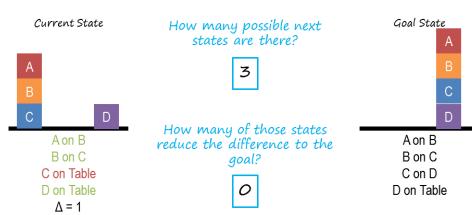


Figure 148: Exercise Block Problem V

That's right David and that means that means-ends analysis doesn't not always take us to what's the goal. Sometimes it can take us away from the goal. And sometimes means-end analysis can get caught in loops. Means-end analysis, like genetic and test, is an example of universal error methods. These universal error methods are applicable to very large classes of problems. However, they can rate few guarantees of success, and they're often very costly. They're costly in terms of computational efficiency. They neither provide any guarantees of computational efficiency, nor provide any guarantees of the optimality of the solution that they come up with. Their power lies in the fact that they can be applied to a very large class of problems. Later in this class, we'll discuss problem-solving methods, which are very specialized problem-solving methods. Those methods are applicable to a smaller class of problems. However, they are more tuned to those problems and often are more efficient and sometimes, also provide guarantees over the optimality of the solution. Although means-end analysis did not work very well for this problem. It in fact works quite well for many other problems and therefore is an important AI method. Later in this class when we come to planning, we will look at more powerful specialized methods that can in fact address this class of problems quite well.

17 - Assignment Means-Ends Analysis

[Click here to watch the video](#)

Assignment

How would you use means-ends analysis to design an agent that could answer Raven's Progressive Matrices?

Figure 149: Assignment Means-Ends Analysis

So how do you use means ends analysis to solve Raven's Progressive Matrices? What exactly is our goal in this context? You might think of the goal in different ways. We might think of it as, the goal is to solve the problem or in a different sense we might think of the goal as the transform sum frame into another frame. And then trace back and find what the transformation was? In that context how would you then measure distance? We noticed that distance is important in doing means ends analysis because that helps us decide what to do next. Once you have a measure of how to actually measure distance to your goal what are the individual operators or moves that you can take to actually move closer to your goal and how would you weight them to be able to decide what to do at any given time. In addition, what are the overall strengths of using means and analysis as a problem solving approach in this context, and what are its limitations. Is it well suited for these problems, or are there perhaps other things that we can be doing that aren't necessarily under this topic that would actually make the problem even easier.

18 - Problem Reduction

[Click here to watch the video](#)

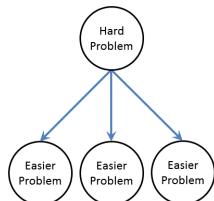


Figure 150: Problem Reduction

Let us now turn to the third problem solving method under this topic called problem reduction. The method of problem reduction actually is quite intuitive. I'm sure you use it all the time. Given the hard complex problem, reduce it. Decompose it into multiple easier, smaller, simpler problems. Consider, for example, computer programming or software design that I'm sure many of you do all the time. Given a hard part of the address, you decompose it with a series of smaller problems. How do I read the input? How do I process it? How do I write the output? That itself is a decomposition. In fact, one of the fundamental roles that knowledge plays is it tells you how to decompose a hard problem into simpler problems. Then once you have solutions to this simpler smaller problems. You can think about how to compose the sub-solutions to the sub-problems into a solution of the problem as a whole. That's how problem reduction works.

19 - Problem Reduction in the Block Problem

[Click here to watch the video](#)



Figure 151: Problem Reduction in the Block Problem

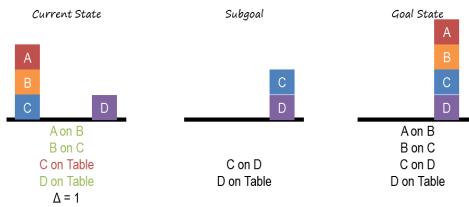


Figure 152: Problem Reduction in the Block Problem



Figure 153: Problem Reduction in the Block Problem

Let us start from where we left off when we finished means ends analysis analysis. This was the current state, this was the goal state. As we saw from means ends analysis analysis, achieving this goal state is not a very easy problem. However, we can think of this goal state as being composed of several sub goals, so D on top of table. C on top of D. B on top of C. A on top of B. Four sub goals here. Now, we can try to address this problem by looking at one sub goal at a time. Let us suppose that we have picked this sub goal, C on top of D. Give that sub goal, we can now start from this current state and try to achieve this sub goal. Now of course, one might ask the question, why did we pick the goal C over D, and not the goal, B over C, or the goal A over B? Well one reason is that, the difference between this state and that state had to do with C over D. But in general, problem reduction by itself does not tell us, what sub-goal to attack first. That is a problem, we'll address later when we come to planning. Well now the major point is, that we can decompose the goal into several subgoals, and attack one subgoal at

a time. Now that we have C over D as a subgoal, we really don't carry about whether A is on B or B is on C. What we are focused on is the other two states, C on table, D on table, because those are the blocks that occur in the goal state. So let us now see how means ends analysis have been solved this sub problem means ends analysis goal C on D and D on Table.

20 - Exercise Problem Reduction I

[Click here to watch the video](#)

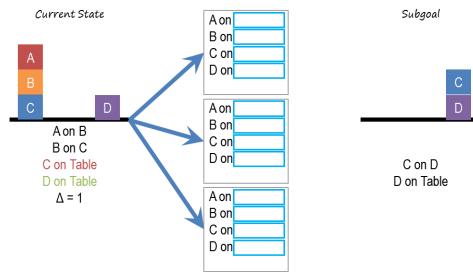


Figure 154: Exercise Problem Reduction I

So given this is a current state, what successor states are possible if we were to apply means and analysis? Please fill in these boxes.

21 - Exercise Problem Reduction I

[Click here to watch the video](#)

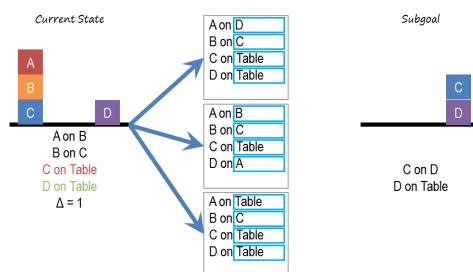


Figure 155: Exercise Problem Reduction I

David, how did you fill up these boxes? So there are three possible moves here. A on the table, A on D, or D on A. So up top I have A on D, moving A to D. In the middle I have moving D up to A, and on the bottom I have A on the table. That looks right, David.

22 - Exercise Problem Reduction II

[Click here to watch the video](#)

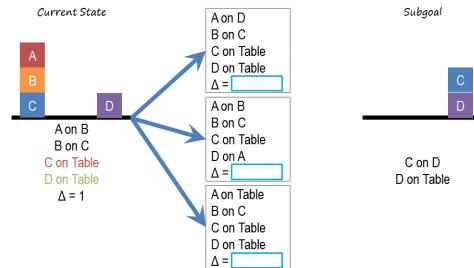


Figure 156: Exercise Problem Reduction II

Let us now calculate the difference from each of the states to the goal state.

23 - Exercise Problem Reduction II

[Click here to watch the video](#)

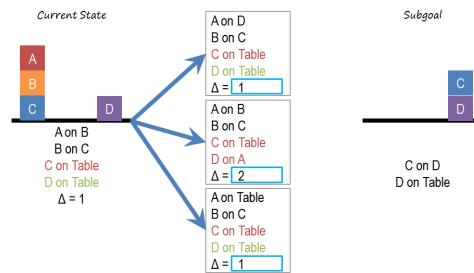


Figure 157: Exercise Problem Reduction II

So note that both the state at the top and this state at the bottom have an equal amount of difference compared to goal state. We could've chosen either state to go further. For now, we're going to go with the one at the bottom. The reason of course is that if I put A on D that will get in the way of solving the rest of the problem. For now, let us go with this state. Later on we will see how an AI agent will decide that this is not a good path to take and this is the better path to take.

24 - Exercise Problem Reduction III

[Click here to watch the video](#)

LESSON 05 - MEANS-ENDS ANALYSIS

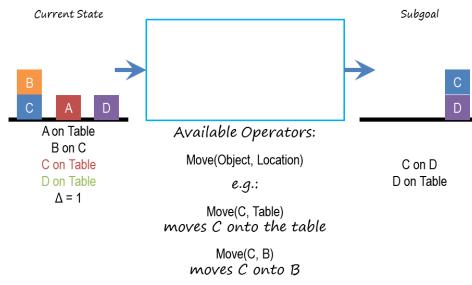


Figure 158: Exercise Problem Reduction III

So if we make the move that we had at the end of the last shot, we'll get this state. So now we need to go from this state to the goal state. Please write down what is the sequence of operators which might take us from the current state to the goal state.

25 - Exercise Problem Reduction III

[Click here to watch the video](#)

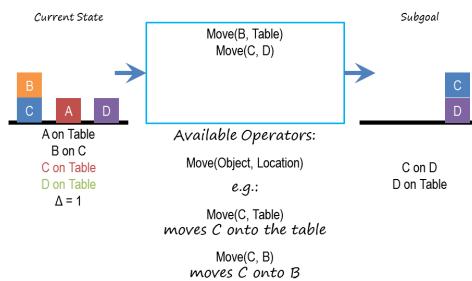


Figure 159: Exercise Problem Reduction III

What sequence did you come up with, David? So from this state, there's seven possible moves we can make initially. Some of them are going to move us away from our goal state. So putting D on A or D on B are going to move us away from our goal state. To means and to analysis, the rest of the moves are relatively the same. Putting A on B doesn't get us closer, putting B on C doesn't get us closer, and putting B on D doesn't get us closer. However, we can see that we really need to get B out of the way of C to move it up here. Like Ashad mentioned, later on we'll talk about how an agent would decide that it needs to get B out of the way of C. But for right now, let's just go ahead and choose B on Table as the next move. Given that, we now

see that of the next possible states, only one reduces our difference, and that's to put C up on D. That was the right answer, David. Thank you. You will note that we're leaving several questions unanswered for now, and that is fine. But you will also note that this round reduction helps us make progress towards solving the problem.

26 - Exercise Problem Reduction III

[Click here to watch the video](#)

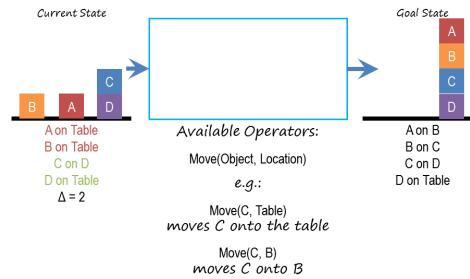


Figure 160: Exercise Problem Reduction III

So the application of the last move in the previous shot will bring us to this state. In this state the the sub-goal C over D has been achieved. Now that we've achieved the first sub-goal, we can worry about achieving the other sub-goals. The other sub-goals, recall, were B over C and A over B. Given this as the current state and this as the goal state. Please write down the sequence of operations that will take us from the current state to the goal state.

27 - Exercise Problem Reduction III

[Click here to watch the video](#)

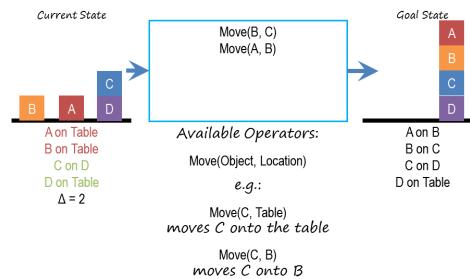


Figure 161: Exercise Problem Reduction III

That was correct, David. Now this particular problem might look very simple. Because for you and me as humans, going from this state

to this state is almost trivial. But notice how many different questions arose in trying to analyze this problem. Clearly, you and I as humans must be addressing these issues. This kind of A.I. analysis makes explicit what is usually tacit when humans solve this problem. And that is one of the powers of A.I.. Indeed we have left a lot of questions unanswered. But each unanswered question then requires an answer. Now we know that if you must develop methods that somehow will help to address those questions. Like genetic [x] tests and like [x] dialysis. Problem reduction is a universal method. It is applicable very large class of problems. Once again, problem reduction does not provide guarantee of successes.

28 - Means-Ends Analysis for Ravens

[Click here to watch the video](#)



Figure 162: Means-Ends Analysis for Ravens

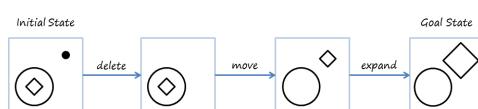


Figure 163: Means-Ends Analysis for Ravens

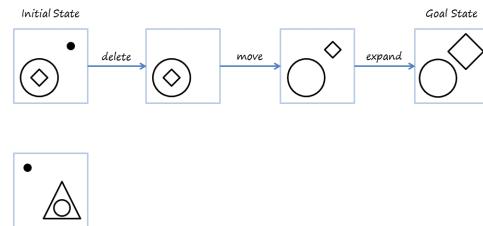


Figure 164: Means-Ends Analysis for Ravens

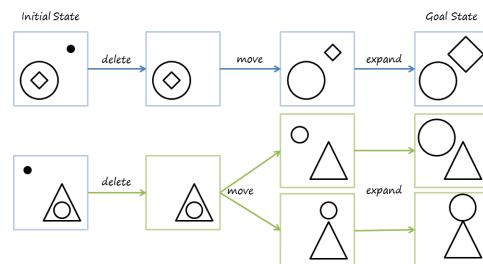


Figure 165: Means-Ends Analysis for Ravens

Now that we have discussed means and analysis in problem production, let's revisit one of the problems that we would encounter when we were talking about the Raven's test of intelligence. So imagine that this is A and this is B. We can think of this as an initial state and this as a goal state. We know that A has been transformed into B. Now we can think in terms of a sequence of operations, that will transform this initial state into the goal state. It is one sequence. Delete it out, then move the diamond out of the circle. Then expanded out. Let us now see, we can think of this as an outward or means of analysis because each move here, brings us a little closer to the goal state. The advantage of doing this analysis and coming up with a sequence of transformations that will take us from the initial state to goal state as that. We cannot in where we are worried about our sub- of applying the same set of transformations to the image free, let's do that. We'll apply one transformation at a time, so we'll apply delete. We delete the dot. Now we apply move to this state, and it can give rise to several states. We have shown two here. Both of these spheres fulfill the requirements of this move operation of taking the

diamond outside the circle. And now for each of the states, we can apply the operation of expand. Here we are expanding the circle, here too we are expanding the circle, although with different amounts. Once again the question here is, what in the image A corresponds to what in image C? How do we know that the diamond inside here corresponds to the circle inside the triangle here? We'll discuss this in detail when we discuss analogical reasoning, but for now, here is a partial answer. Remember that we had a semantic network representation of the image A. And in that representation, we said that the diamond is inside the circle. Now, we also have a semantic network representation of image C. And that semantic network representation says that the circle is inside the triangle. But it's that inside relationship that hints that this circle must correspond to the diamond. Because here the diamond is inside and here the circle is inside. So note then, that their presentation allows us to answer several questions about similarity, about correspondence and when we have metals, like means and analysis then we can do a systematic analysis of this transformations, and try to transfer this set of transformations to the new problem. What's interesting is we can also see this as an example of problem reduction. We initially just had a big problem that we had to solve. But here, we've reduced it to three subgoals. Our first subgoal is to find the transformation between A and B. Our second subgoal, is to transfer that transformation to C and find some candidate states for D. And our third subgoal, would be to compare those candidate states for D to each of the choices in our problem. That's good analysis, David. Let's go one step further. This also has generated. We are generating solutions that we can then test against the various choices that were given to us. So in this particular problem, you can see means and analysis working, problem reduction working, and the Raven's test working. Often, solving the complex problem requires a combination of AI techniques. At one point, one might use problem reduction. At another point, one might use the Raven's test, and at a third point, one might use means and analysis. Notice also, that the

one single knowledge representation of semantic network supports all three of these strategies. The coupling between the knowledge representation of semantic network, and any of these three strategies, problem reduction, means and analysis, and generating test is weak. Later on we'll come across methods in which knowledge and the problem solving method are closely coupled. The knowledge affords certain inferences, and inferences demand certain kinds of knowledge. This is why these methods are known as Weak Methods. Because the coupling between this universal method and the knowledge representation is weak.

29 - Assignment Problem Reduction

[Click here to watch the video](#)

Assignment

How would you use problem reduction to design an agent that could answer Raven's Progressive Matrices?

Figure 166: Assignment Problem Reduction

So how would you apply a problem reduction to Raven's Progressive Matrices? Before we actually talk about how our agents would do it, we can think about how we would do it. When we are solving a matrix, where do the smaller or easier problems that we are actually breaking it down into? How are we solving those smaller problems, and how are we then combining them into an answer to the problem as a whole? Once we know how we're doing it, how will your agent actually be able to do the same kind of reasoning process? How will it recognize when to split a problem in to smaller problems? How will it solve the smaller problems? And how will it then combine those in to an answer to the problem as whole? During this process think about, what exactly is it that makes these smaller problems easier for your agent to answer than just answering the problem as a whole? And how does that actually help you solve these problems better?

30 - Wrap Up[Click here to watch the video](#)*To recap...*

- State spaces
- Means-ends analysis
- Problem solving with means-ends analysis
- Problem reduction

Figure 167: Wrap Up

So let's wrap up what we've talked about today. We started off today by talking about state spaces and we used this to frame our discussion of mean-ends analysis. Means-ends analysis is a very general purpose problem solving method, that allows us to look at our goal and try to continually move towards it. We then use means-ends analysis to try and address a couple of different kinds of problems. But when we did so, we hit an obstacle. To overcome that obstacle, we used problem reduction. We can use problem reduction in a lot of other problem solving contexts, but here we use it to specifically to overcome the obstacle we hit during means-ends analysis. Problem reduction occurs and we take a big hard problem and introduce it into smaller easier problems. By solving the smaller easier problems, we solve the big hard problem. Next time we're going to talk about production systems, which are the last part of the fundamental areas of our course. But if you're particularly interested in what we've talked about today, you may wish to jump forward to logic and planning. Those were built specifically on the types of the problems we talked about today. And in fact in planning, we'll see a more robust way of solving the kinds of obstacles that we hit, during our exercise with means and analysis earlier in this lesson.

31 - The Cognitive Connection[Click here to watch the video](#)

Let us examine the connection between methods like means ends analysis and problem

reduction on one hand, and human cognition on the other. Methods like means ends analysis, problem reduction and even generate and test, are sometimes called weak methods. They are weak because they make only little use of knowledge. Later on, we'll look at strong methods that are knowledge intensive. That will demand a lot of knowledge. The good thing about those knowledge intensive methods is, that they will actually use knowledge about the world, to come up with good solutions in an efficient manner. On the other hand, those knowledge intensive methods require knowledge, which is not always available. So humans, when they are working in a domain, in a world at which they are experts, tend to use those knowledge intensive methods because they know a lot about the world. But of course, you and I constantly work in worlds, in domains in which we are not experts. When we're not an expert in our domain, a domain that might be unfamiliar to us, then we might well go with matters that are weak because they don't require a lot of knowledge.

32 - Final Quiz[Click here to watch the video](#)

We're at the end of this lesson. Please summarize what you learned in this lesson, inside this box.

33 - Final Quiz[Click here to watch the video](#)

And thank you for doing it.

Summary

This lesson cover the following topics:

1. The knowledge representation of Semantic networks works well with Generate and Test, Means-Ends Analysis and Problem Reduction. These are examples of Universal AI methods.
2. Means-ends analysis uses a heuristic to guide the search from the initial state to the goal state. Convergence is not guaranteed.

Optimality is not guaranteed. Computational efficiency is not guaranteed.

3. Problem reduction is used along with Means-ends analysis to help overcome problems with means-ends analysis.

4. The Universal methods have weak coupling between the methods and the knowledge representation and hence are called Weak methods since they make little use of knowledge. Strong AI methods are knowledge-intensive and use knowledge of the world to come up with good solutions in an efficient manner.

References

1. Winston P., Artificial Intelligence, Chapter 3.

Optional Reading:

1. Winston Chapter 3, pp. 50-60; [Click here](#)

Exercises

Exercise 1:

The workers installed a new wall-to-wall carpet in your bedroom and left. Later you discover that the bedroom door will not close unless you trim off about 1/8 inch off the carpet at the bottom of the door. Construct a difference-operator table relating simple instruments such as a saw, a plane, and a file. Now show how the method of mean-ends

analysis would use the difference-operator table to address the carpet problem in your bedroom.

Exercise 2:

In the Monkey & Bananas problem, a monkey is faced with the problem of reaching bananas hanging from the ceiling. But a box is available that will enable the monkey to reach the bananas if he climbs on it. Initially the monkey is at location A, bananas at B, and the box at C. The bananas are at height Y, the monkey and the box have height X such that if the monkey climbs on the box, it too will be at height Y.

Invent operators Walk, Push, Climb, and Grasp, for walking to a location, pushing the box to a location, climbing up on the box, and grasping bananas, respectively. Show the preconditions and postconditions of each operator.

Write down the initial and final states in propositional form.

Show how the monkey may used means-ends analysis to form a plan to get the bananas in the above problem.

Lesson 06 - Production Systems



Any problem that can be solved by your in-house expert in a 10-30 minute telephone call can be developed as an expert system.
— M. Firebaugh, *Artificial Intelligence: A Knowledge-Based Approach*.

01 - Preview

[Click here to watch the video](#)

Fundamentals

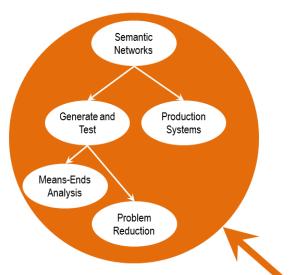


Figure 168: Preview

kind of cognitive architecture, in which knowledge is represented in the form of rules. This is the last topic under the fundamental topics part of the course. We'll start by talking about cognitive architectures in general, then focus on production systems, then come to learning, a particular mechanism of learning called chunking.

02 - Exercise A Pitcher

[Click here to watch the video](#)



Figure 170: Exercise A Pitcher

Lesson Preview

- Cognitive architectures
- Production systems
- Chunking

Figure 169: Preview

Today, we'll talk about production systems. I think you're going to enjoy this, because part of production systems is going to do with learning. And this is the first time in the course we'll be talking about learning. Production systems are

It's the top of the 7th inning. There are runners on 2nd and 3rd base. There are two outs. The batter, Martin Prado, has an average of .256 and bats fourth in the batting order. We are winning 3-2. I struck this batter out last time. My goal is to escape the inning.

What should the pitcher do?

- o Pitch to the batter
- o Intentionally walk the batter

Figure 171: Exercise A Pitcher

To illustrate production systems, let us imagine that you are a baseball pitcher. This illustration is coming from a game between Atlanta Braves and the Arizona Diamondbacks. Here is a pitcher on the mound, and a pitcher has to decide, whether to pitch a ball to the batter, or whether to walk the batter. To be more specific, take a look at the story here in the left. Read it. And to decide what would the pitcher do. What would you do? What would an intelligent agent do? If you don't know much about baseball, don't worry about it. Part of the goal here is to see what someone who does not know about, a lot about baseball may do.

03 - Exercise A Pitcher

[Click here to watch the video](#)

It's the top of the 7th inning. There are runners on 2nd and 3rd base. There are two outs. The batter, Martin Prado, has an average of .256 and bats fourth in the batting order. We are winning 3-2. I struck this batter out last time. My goal is to escape the inning.

- What should the pitcher do?
- o Pitch to the batter
 - Intentionally walk the batter

Figure 172: Exercise A Pitcher

David, it's clear that you know more about baseball than I do. So I assume that your answer is the right one. But notice what is happening here. David has a lot of knowledge about baseball, and he's using that knowledge to make a decision. How is he using his knowledge to make a decision? What is the architecture? What is the reasoning that leads him to make that specific decision? This is one of the things we'll learn in this lesson. How might intelligent agents make complex decisions about the world?

04 - Function of a Cognitive Architecture

[Click here to watch the video](#)

Function for cognitive architectures:

$$f: P^* \rightarrow A$$

Percepts \rightarrow Action

Figure 173: Function of a Cognitive Architecture

Before we talk about cognitive architectures in more detail, first let us characterize what is a cognitive agent. Now, there are many views of what a cognitive agent is. Here is one characterization of cognitive agent which is very popular in modern AI. A cognitive agent is a function that maps a perceptual history into an action. So it's B asterisk map into A where the asterisk on P stands for the history percepts. So this characterization says that one of the major tasks of cognitive agents is to select actions. You're driving a car. What action should you take next? You're having lunch, what action should you do next? You're conversing with someone, someone else, what action should you do next? All the time we are taking actions, we as humans. And the question is, what should we do next? We base that action in part on the history of percepts in that particular situation. So this characterization captures the basic notion of cognitive agent in a very concise manner.

05 - Levels of Cognitive Architectures

[Click here to watch the video](#)

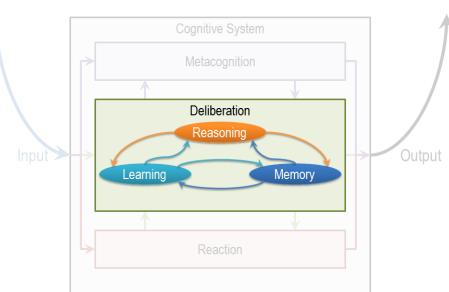


Figure 174: Levels of Cognitive Architectures

We can build queries of knowledge based AI at many levels of instructions. This is the scale

here, low level to high level. At one level, we can build queries at hardware level. So we can talk about a brain or transistor sort of microchip. At the next level, we can talk about the kinds of methods and the kinds of representations we have been talking about, means-ends analysis that has an algorithm associated with it, or semantic network that's a knowledge representation in some symbolic form. At a yet higher level, we can talk about knowledge and tasks. So, the question here becomes what exactly is the task the decision maker has to make? What exactly is the knowledge the decision maker has? So, when David was giving his answer about what the pitcher might do in the situation I showed you earlier, David was clearly using a lot of knowledge, and he was trying to use this knowledge toward to a particular task. Now, in the history of AI, David Mark talked about three levels, the level of tasks, which he called the [computational] theory, the level of algorithms and the level of implementation. And [computational] talked also about multiple levels. He was talking about the knowledge level, the symbol level and lower levels like the hardware. The various levels are connected with each other. So I might think that, the hardware level is a level for implementing what is happening at the algorithm level. And the algorithm level provides and architecture for implementing what is happening at the task level. In the opposite direction, I might think that the task level provides the content of what needs to be represented or manipulated at the algorithm level. And the algorithm and symbol level provide the content for what needs to be manipulated at processor, the hardware level. So as an example, one might say, we're representing this in a form of a semantic network, fair enough. But, what exactly are you going to represent in a semantic network? That's going to come from the knowledge level. It is the knowledge level that tells us, what is the content of the knowledge that is required to play baseball? Once you have the content of knowledge, you can perhaps implement it in many different ways. One way is through semantic network. Similarly, once you know what kind of decision you have to make, and what a decision-making

process might look like overall, there might be many different methods of making that particular decision. Just like we can build a relationship between the task level and the algorithm level, a similar relationship exists between the algorithm level and the hardware level. This is an important point, so let me take another example. All of you are familiar with your standard smartphone. Let us suppose that I was coming from Mars, I was a Martian. And I did not know how your mobile phone works. So I would ask you, well, how, exactly, does your mobile phone work? And you might give an account of how the phone works at that level of distraction. There will be a legitimate account, you have somewhere else for give an account of how the smartphone works at a, at a level of tasks and knowledge. This person might say, well, a phone allows you to communicate with other people at long distances. How that is implemented is a different matter. Now you will see, I'm sure, that all three of these interpretations, all three of these descriptions are legitimate and valid. You will see also, that we really need all three of these levels of description. We do need to understand what this smartphone does, and that kind of knowledge it uses to do it. We do need to understand what kind of algorithm and knowledge presentations it uses, and what kind of hardware implements all of this. Now, you can do a similar kind of analysis for other kinds of devices. Let's say, like your calculator. Could we do a similar kind of analysis for intelligent agents? Are these different layers also meaningful for analyzing what happens in cognitive systems, whether they are natural or artificial? And at what layer should we be building a theory? Our hypothesis is, that these three layers are also useful for trying to analyze how cognitive systems might work, but natural cognitive systems and artificial cognitive systems. Further, our hypothesis is that we want to build theories at all three of these different level, levels of abstraction, not at any one of them. In fact, their constraint's flowing in both directions. If we know about what kind of tasks we want to do and what kind of knowledge we want to use, then that tells us something about what kind of algorithms we need to

and what kind of knowledge representations we need. And that tells us something about what kind of hardware we need. In the other direction, if we know what kind of hardware we have that imposes constraints and provide [computational] for what kind algorithms and knowledge representations can be there, which then provides accordance within constraints. Well, what kinds of tasks can be done and what kind of knowledge can be used. In this class, we'll be concerned mostly with the top two layers, although I allude occasionally to the third layer as well. A lot of work in AI is at the top two layers of abstraction.

06 - Exercise Levels of Architectures

[Click here to watch the video](#)

What are the layers of Watson?



Figure 175: Exercise Levels of Architectures

Now we have talked about Watson as a possible example for cognitive system earlier. And now we have talked about various layers of abstraction at which we can analyze a cognitive system. So what do you think are the layers of analysis of Watson?

07 - Exercise Levels of Architectures

[Click here to watch the video](#)

What are the layers of Watson?

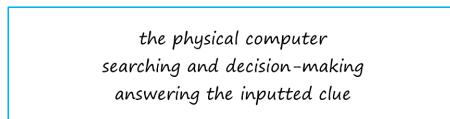


Figure 176: Exercise Levels of Architectures

Derrick what do you think about this? So what we talked about before was that the bottom layer is the hardware layer. I'm not completely sure what the hardware of Watson looks like but I assume there has to be a physical computer there somewhere. At the next layer, the algorithm layer, Watson has some general searching and decision-making ability that allows him to search over his knowledge base and come to decisions. These are general abilities that are separated from whatever individual thing he's trying to think about right now. And then at the top layer we can think about exactly that, what he's trying to think about right now. So the task here might be answering the inputted clue. This would involve searching on the individual information in the clue, processing his knowledge specifically about that data, and then coming to an answer specifically for that clue. That was a good answer David. So the again the three layers and note that in the task layer here, answering the inputted clue, knowledge is also a part of it. Knowledge that Watson must have in order to be able to answer that particular question, how that knowledge is implemented, what kind of presentations it uses goes in the second layer.

08 - Assumptions of Cognitive Architectures

[Click here to watch the video](#)

Assumptions of a Cognitive Architecture

- Goal-oriented
- Rich, complex environment
- Significant knowledge
- Symbols and abstractions
- Flexible and function of the environment
- Learning

Figure 177: Assumptions of Cognitive Architectures

The school of AI that works on cognitive architectures makes sort of fundamental assumptions about the nature of cognitive agents. First, that cognitive agents are goal oriented, or goal directed. They have goals and they take actions in the pursuit of those goals. Second, that

these cognitive agents live in a rich, complex, dynamic environments. Third, this cognitive agent used knowledge of the world in order to pursue their goals in this rich complex dynamic environments. Fourth, that this knowledge is particular abstraction that captures the important things about the world that the level of abstraction and removes all the details. And at that level of abstraction, knowledge is captured in the form of symbols. Fifth, the cognitive agents are very flexible. The behavior is dependent upon the environment. As environment changes, so does the behavior. And sixth cognitive agents learn from their experiences. They're constantly learning as they interact with the world.

09 - Architecture Content Behavior

[Click here to watch the video](#)

$$\begin{array}{rcl} \text{Architecture} & + & \\ & \text{Content} & \\ & = & \\ & \text{Behavior} & \end{array}$$

Figure 178: Architecture Content Behavior

order to do that, you have to design the right architecture, and then put the right kind of knowledge content into that architecture, to get the behavior that you want from it. That's a complicated thing. But suppose that I could fix the architecture for you. In that case, if the architecture is fixed, I simply have to change the knowledge content to get different behaviors, which is a really powerful idea. From a different direction, suppose that we were trying to understand human behavior. Now we could say, again, that the architecture is fixed, we could say that, this behavior is arising because the knowledge content is different. We can map now, behavior to content because the architecture is fixed. That simplifies our understanding of how to design machines or how to understand human cognition. By the way, the same thing happens in computer architecture. I'm sure you have, are familiar with computer architecture. Computer architecture has stored programs in it, that's the content, and that running of the stored program gives you different behaviors. The computer architecture doesn't change, the stored program keeps on changing, to give you different kind of behaviors. Same idea with cognitive architectures. Keep the architecture constant, change the content. Now, of course, the big question will become, what is a good architecture? And that's what we'll examine later.

Function for cognitive architectures:

$$f: P^* \rightarrow A$$

Percepts \rightarrow Action

Figure 179: Architecture Content Behavior

We can capture the basic intuition behind work on cognitive architectures by a simple equation, architecture plus content equals behavior. Let us look at this equation from two different perspectives. First, imagine that you want to design an intelligent machine that exhibits a particular kind of behavior. This equation says that, in

10 - A Cognitive Architecture for Production Systems

[Click here to watch the video](#)

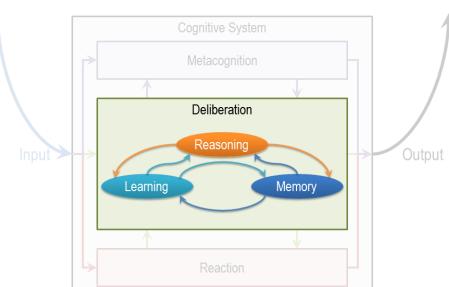


Figure 180: A Cognitive Architecture for Production Systems

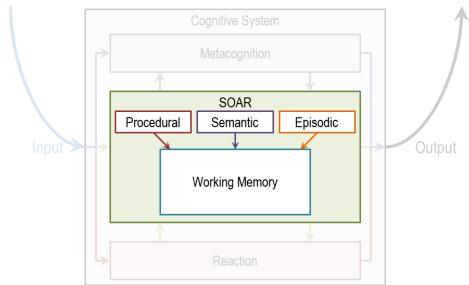


Figure 181: A Cognitive Architecture for Production Systems

So we have come across this high level architecture for deliberation earlier. Today we will talk about a specific cognitive architecture for deliberation. This architecture is called SOAR. I should mention that SOAR not only covers deliberation, SOAR can also cover certain aspects of reaction, and some aspects of meta cognition. But we are going to focus mostly on the deliberation component and so on. SOAR was initiated by Allen Neville, John Lear, and Paul Rosenbloom. And John Lear and Paul Rosenbloom have been working on it for the last 30 years or so. The highest level consists of a long term memory and a working memory. The [David] itself contains different kinds of knowledge. In particular SOAR talks about three kinds of knowledge. Procedural, semantic, and episodic. Episodic knowledge has to do with events. Specific instances of events, like, what did you have for dinner yesterday. Semantic knowledge has to do with generalizations in the form of concepts and models of the world. For example, your concept of a human being, or your model of how a plane flies in the air. Procedural has to do with how to do certain things. So as an example, how do you pour water from a jug into a tumbler. Notice that this makes an architecture. There are different components that are interacting with each other. This arrangement of components will afford certain processes of reasoning and learning. That's exactly the kind of processes of reasoning and learning that we'll look at next.

11 - Return to the Pitcher

[Click here to watch the video](#)

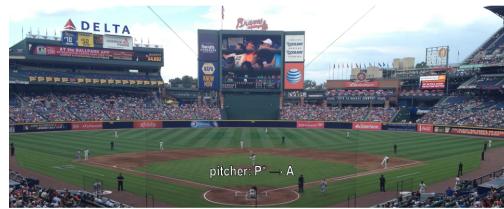


Figure 182: Return to the Pitcher

It's the top of the 7th inning. There are runners on 2nd and 3rd base. There are two outs. The batter, Martin Prado, has an average of .256 and bats fourth in the batting order. We are winning 3-2. I struck this batter out last time. My goal is to escape the inning.

- What should the pitcher do?
 - o Pitch to the batter
 - Intentionally walk the batter

Figure 183: Return to the Pitcher



Figure 184: Return to the Pitcher

Let us now go back to example of the baseball pitcher who has to decide on a action to take in a particular circumstance. So we can think of this pitcher as mapping a percept history into an action. Now imagine that this pitcher is embodying a production system. We are back to a very specific situation, and you can certainly read it again. Recall that David had given the answer, the pitcher will intentionally walk the batter. So we want to make the theory of how might the pitcher or internal and AI agent come to this decision. Recall the very specific situation that the pitcher is facing. And recall also that David had come up with this answer. So, here is a set of percepts, and here is an action. And

the question is, how these percepts get mapped into this action? We are going to add one build a theory of how the human pitcher might be making these decisions, as well as a theory of how an AI agent could be built to make this decision. So let's go back to the example of the pitcher having to decide on a action in a particular situation in the world. So the pitcher has several kinds of knowledge. Some of its knowledge is internal. It already has it. Some of it, it can perceive from the world around it. As an example, the pitcher can perceive the various objects here, such as the bases, first, second, third base. The pitcher can perceive the batter here. The pitcher can perceive the current state of the game. The specific score in the inning, the specific batter. The pitcher can perceive the positions of its own teammates. So, all these things the pitcher can perceive, and these then are become exact specific kinds of knowledge that each pitcher has. The pitcher also has internal knowledge. The pitcher has knowledge about his goals and objectives here.

12 - Action Selection

[Click here to watch the video](#)

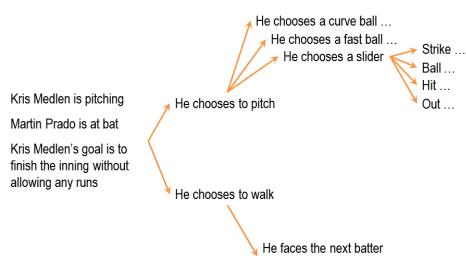


Figure 185: Action Selection

So imagine that Kris Medlen from Atlanta Braves is the pitcher. And Martin Prado from Arizona Diamondbacks is at the bat. Kris Medlen has the goal of finishing the inning without allowing any runs. How does Kris Medlen decide on an action? We may conceptualize Medlen's decision making like the following. Medlen may look at various choices that are available to him. He may throw a pitch. Or he may choose to walk the batter. If he walks the batter, then there are additional possibilities that open up. He'll need to face the next batter. If he chooses to pitch, then he'll have to decide what kind of ball to throw. A slider, a fast ball, or a curve ball. If it was a slider, then there is a next set of possibilities open up. There might be a strike or a ball or a hit or he may just strike the batter out. Thus, Medlen is setting up a state space. Now, what we just did informally can be stated formally. So, we can imagine a number of states in the states space. The state space is a combination of all the states that can be achieved by applying various combinations of operators, starting from the initial state. Each state can be described in terms of some features, f_1, f_2 , there could be more. Each feature can take on some values. For example, v_1 , there might be a range of values here. So initially, the picture is at state s_0 . And the pitcher wants to assume some state s_{101} . And at a state s_{101} presumably the pitcher the pitcher's goal has been accomplished. So we may think as the pitcher's decision making as some kind of a part of its current state to this particular goal state. This is an abstract space. The pitcher has not yet made any action. The picture is still thinking. The picture is sitting up an abstract state space in his mind and exploding that state space.

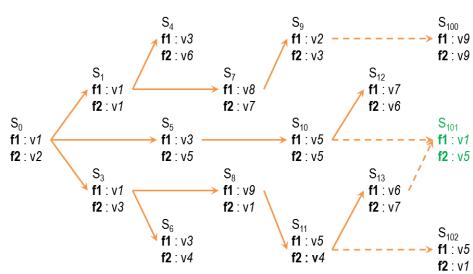


Figure 186: Action Selection

13 - Putting Content in the Architecture

[Click here to watch the video](#)

It's the top of the 7th inning. There are runners on 2nd and 3rd base. There are two outs. The batter, Martin Prado, has an average of .256 and bats fourth in the batting order. We are winning 3-2. I struck this batter out last time. My goal is to escape the inning.

```

inning : 7th
portion : top
runners : 2nd and 3rd
outs : 2
batter : Prado
average : .256
bats : right-handed
score : 3-2
goal : escape inning
    
```

Figure 187: Putting Content in the Architecture

Okay, now in order to go further, let us start thinking in terms of, how we can put all of these precepts and goal, into some feature value language, so that we can store it inside Sole. It is one attempt at capturing all of this knowledge, so I can say that it's the 7th inning. Inning is 7th. It's the top of the 7th inning. It's the top here. Runners are on 2nd and 3rd base. 2nd and 3rd base. And then so on and so forth. Note that at the bottom I have goal is to escape the inning. Which I think means in this particular context, to go to the next inning, without letting the batter score anymore points. So now that we have. Put all of this precepts coming from the world and the goal, into some kind of simple representation which has features and values in it, the fun is going to begin.

14 - Bringing in Memory
[Click here to watch the video](#)

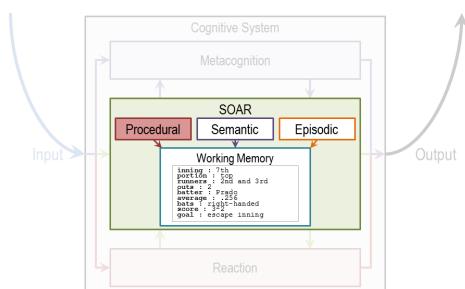


Figure 188: Bringing in Memory

```

inning : 7th
portion : top
runners : 2nd and 3rd
outs : 2
batter : Prado
average : .256
bats : right-handed
score : 3-2
goal : escape inning

(r1) If goal is to Escape, I perceive 2 outs, I perceive a runner on 2nd and I perceive no runner on 1st
then suggest goal/intentionally walk batter/intentional/Walk

(r2) If goal is to Escape I perceive fewer than 2 outs, or I perceive a runner on 1st, or I perceive no runner on 2nd, or I perceive no runners
then suggest goal to get the batter out via pitching (Pitch)

(r3) If goal is intentional Walk
then suggest intentional-walk operator

(r4) If goal is Pitch and I perceive a new batter who is left/right-handed
then add batter not out, balls 0, strikes 0, bats left/right

(r5) If the goal is Pitch and batter not out
then suggest throw-curve-ball operator

(r6) If the goal is Pitch and batter not out and bats left-handed
then suggest throw-fast-ball operator

(r7) If only one operator has been selected
then send operator to the motor system and add pitch thrown to state
    ... ...
    
```

Figure 189: Bringing in Memory

So, source working memory not contains all the things that we had in the previous shot. Some of these are percepts. Some the things are the pitcher's internal goals. Let us see how the contents of the working memory, now invoke different kinds of knowledge from the long term memory. So, let us imagine that the procedural part of source long term memory contains the following rules. The procedural knowledge and source long term memory is represented in the form of rules. The system sometimes called production group. In fact, the term production systems comes from the term production group. So, each rule here is a production group. I've shown seven here, there could be more rules. Once again, these production rules are captured in the procedural knowledge, and soource long-term memory. Recall that one of the first things that the pitcher had to decide was whether the pitcher should throw a pitch or walk the batter. Therefore, we assume that there are some rules which allow the pitcher to make a decision between these two choices. Thus there is a rule here. If the goal is to escape, and I perceive two outs, and I perceive on the second, and I perceive a runner on the second and I perceive no runner on the first base, then I'm suggest a goal of intentionally walk batter. There is another rule. The second rule says, if the goal is to escape, and I perceive two outs and I perceive a runner on the first base. Or if perceived not out on the second, or if perceived no runners, then suggest the goal to get the batter out via pitching. And what might happen if I pick the goal of intentionally walking the batter? The goal is to intentionally walk the batter then suggest intentional walk operator. Now, this intentional

walk operator corresponds to some action available to all. And similarly, the other rules. Let's consider one other rule, rule number seven. If only one operator has been selected, then send the operator to the motor system and add the pitch thrown to the state. So, there is both now an action has been selected, and the action is going to be executed. As well as the state of the working memory is going to be changed, so it will say that now the pitch has been thrown. Before we go ahead, let me summarize what we just learned. Source long term memory consists of various kinds of knowledge. One kind of knowledge, the one that we are considering right now is procedural knowledge. Procedural knowledge is about how to do something. And so procedural knowledge is represented in the form of production rules. Each production rule is, is of the form, if something then something. There are antecedents and their consequence. These antecedents may be connected through various kind of relationships, like and, and or. The consequence too might be connected through various kind of relationships like and, and or. So I may have if some antecedent is true, and some other, other ante, antecedent is true, and so on, then do some consequent. Now that we understand a little bit about the representation of production rules. What about the content? What should be the content that we put into these production rules? Earlier we had said that cognitive architectures are goal oriented. So we'll expect goals to appear in some of the production rules. Indeed, they do. R1, r2, r3, and so on. Earlier we had said that knowledge based AI cognitive systems, use a lot of knowledge. And you can see how detailed and specific this knowledge is. In fact, the knowledge is so detailed and specific that in principle we can hope that as different percepts come from the world, some rule is available that will be useful for that set of percepts.

15 - Exercise Production System in Action I
[Click here to watch the video](#)

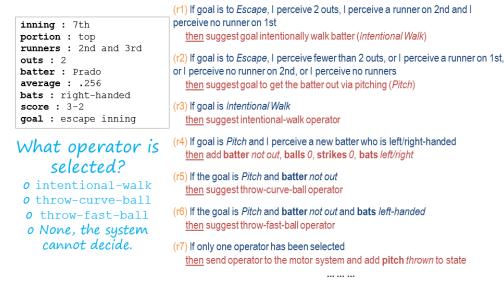


Figure 190: Exercise Production System in Action I

Okay, given these productions in the procedural part of sol's long term memory. And given these are the contents of the working memory which capture the current set of percepts and the current goal. What operator do you think sol will select? Note that one of the choices here is none, the system cannot decide.

16 - Exercise Production System in Action I

[Click here to watch the video](#)

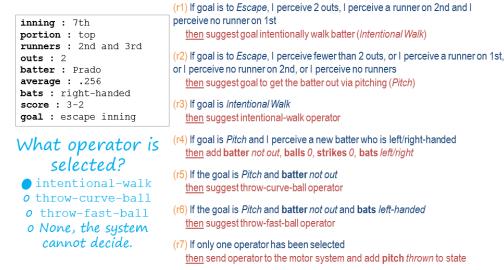


Figure 191: Exercise Production System in Action I

That was right, David. So note what happened. There were the contents of the working memory. Here were all the rules of a level in the procedural part of the long-term memory. And so then, match the contents of the working memory with the antecedents of the various productions. Depending on the match between the contents of the working memory, and the antecedents of the rules, some rules got activated, fired as some people say. Depending upon what rule got fired, that resulted in, perhaps, firing of additional rules. So as David said, if the rule number one got activated because the goal became intentionally walk the batter, that then

lead to the activation of rule number three, which was to select the intentional-walk operator. In this way, given the set of contents of the working memory and a mapping between those contents and the antecedents of the various production rules that capture the procedure knowledge. Some rules get activated, and this activation continues until sort cannot activate any additional rules. At that point, sort has given an answer, based on the consequence of a rule that matches a motor action. Note that sort just provided an account of how the picture decided on a specific page, on a specific action. Note that, we're started with the goal of providing an account of how the pitcher selects on an action or how an AI agent selects on an action. So what is the account? Based on the goal and the percents, match them with the antecedence of the rules that captured the procedural knowledge. And then, accept the consequent of some production that matches some motor action from some precepts we have gone to some action.

17 - Exercise Production System in Action II

[Click here to watch the video](#)

```

inning : 7th
portion : top
runner_s : 1st, 2nd, 3rd
outs : 2
batter : Hill
average : .269
bats : right-handed
score : 3-2
goal : escape inning

What operator is selected?
 intentional-walk
 throw-curve-ball
 throw-fast-ball
 None, the system cannot decide.

(1) If goal is to Escape, I perceive 2 outs, I perceive a runner on 2nd and I perceive no runner on 1st
    then suggest goal intentionally walk batter (Intentional Walk)
(2) If goal is to Escape I perceive fewer than 2 outs, or I perceive a runner on 1st, or I perceive no runner on 2nd, or I perceive no runners
    then suggest goal to get the batter out via pitching (Pitch)
(3) If goal is Intentional Walk
    then suggest intentional-walk operator
(4) If goal is Pitch and I perceive a new batter who is left/right-handed
    then add batter not out, balls 0, strikes 0, bats left/right
(5) If the goal is Pitch and batter not out
    then suggest throw-curve-ball operator
(6) If the goal is Pitch and batter not out and bats left-handed
    then suggest throw-fast-ball operator
(7) If only one operator has been selected
    then send operator to the motor system and add pitch thrown to state
    .... ...

```

Figure 192: Exercise Production System in Action II

Now let us consider another situation. Suppose that our pitcher actually was able to walk the batter. So, now, there are runners on the first, second, and third bases. Not just on the second and third bases, but one on the first, also. So the picture succeeded in accomplishing its goal in the last shot. So the current situation then is discard, but this side of percept and this goal. The confidence of the working memory have just changed. Of course, the production

rules capturing the pursuit of knowledge have not yet changed. So these are exactly the same productions that we had previously. Only the contents of the working memory has changed. We now have a runner at the first as well as the second and the third. And this exercise is very interesting because it will lead us to a different set of conclusions. Given the set of precepts in this goal and the set of production rules, what operator do you think the picture will select?

18 - Exercise Production System in Action II

[Click here to watch the video](#)

```

inning : 7th
portion : top
runner_s : 1st, 2nd, 3rd
outs : 2
batter : Hill
average : .269
bats : right-handed
score : 3-2
goal : escape inning

What operator is selected?
 intentional-walk
 throw-curve-ball
 throw-fast-ball
 None, the system cannot decide.

(1) If goal is to Escape, I perceive 2 outs, I perceive a runner on 2nd and I perceive no runner on 1st
    then suggest goal intentionally walk batter (Intentional Walk)
(2) If goal is to Escape I perceive fewer than 2 outs, or I perceive a runner on 1st, or I perceive no runner on 2nd, or I perceive no runners
    then suggest goal to get the batter out via pitching (Pitch)
(3) If goal is Intentional Walk
    then suggest intentional-walk operator
(4) If goal is Pitch and I perceive a new batter who is left/right-handed
    then add batter not out, balls 0, strikes 0, bats left/right
(5) If the goal is Pitch and batter not out
    then suggest throw-curve-ball operator
(6) If the goal is Pitch and batter not out and bats left-handed
    then suggest throw-fast-ball operator
(7) If only one operator has been selected
    then send operator to the motor system and add pitch thrown to state
    .... ...

```

Figure 193: Exercise Production System in Action II

David, what do you think the patient would select next? So unlike last time, the first production rule won't fire because I do perceive a runner on first. Instead, while I don't perceive fewer than two outs, I do perceive a runner on first. So this production rule will fire, and now by goal is to get the batter out via pitching. That'll skip the next rule and come down to here. Given that the goal is to pitch and we have a new batter, we're going to add some information to our stake. So we can actually now, not just keep track of the game as a whole, but we can keep track of this individual exchange with the batter. Then the goal is to pitch and the batter is not yet out, so we're going to suggest the throw-curve-ball-operator. We then look and see that the batter bats right handed so the next rule doesn't fire. So at this point, we've only suggest one operator, the throw-curve-ball-operator. So we're going to send that to our motor system. So the answer will be that we'll throw a curve ball. That was right, David, thank you. Let's summarize some of the things that David noted.

So, based on the contents of the working memory, some rules get activated. As these rules get activated, some consequence get established. As this consequence get established, they get written. Those consequence get written on the working memory. So the contents of working memory are constantly changing. As the contents of the working memory change, nodules can get activated. So there is a constant interaction between the working memory and the long term memory. The contents of the working memory change quite rapidly. The contents of the long term memory change very, very slowly.

19 - Exercise System in Action III

[Click here to watch the video](#)

```

inning : 7th
portion : top
runners : 1st, 2nd, 3rd
outs : 2
batter : Parra
average : .273
bats : left-handed
score : 3-2
goal : escape inning

What operator is selected?
o intentional-walk
o throw-curve-ball
o throw-fast-ball
o None, the system cannot decide.

(r1) If goal is to Escape I perceive 2 outs, I perceive a runner on 2nd and I
perceive no runner on 1st
then suggest goal intentionally walk batter (Intentional Walk)

(r2) If goal is to Escape I perceive fewer than 2 outs, or I perceive a runner on 1st,
or I perceive no runner on 2nd, or I perceive no runners
then suggest goal to get the batter out via pitching (Pitch)

(r3) If goal is intentional Walk
then suggest intentional-walk operator

(r4) If goal is Pitch and I perceive a new batter who is left/right-handed
then add batter not out, balls 0, strikes 0, bats left/right

(r5) If the goal is Pitch and batter not out
then suggest throw-curve-ball operator

(r6) If the goal is Pitch and batter not out and bats left-handed
then suggest throw-fast-ball operator

(r7) If only one operator has been selected
then send operator to the motor system and add pitch thrown to state
.... ...

```

Figure 194: Exercise System in Action III

Aha, so this situation keeps becoming more complicated David. Let's think about what might happen if the manager of the Arizona Diamondbacks anticipated what the Atlanta Braves pitcher would do, and actually change the batter so the batter now is left handed? If the batter is left handed, then the percept is slightly different. The content of the pitcher's working memory is slightly different. The production rules capturing the pursuit of knowledge are still the same. What do you think will happen now? What kind of decision will the pitcher make now?

20 - Exercise System in Action III

[Click here to watch the video](#)

```

inning : 7th
portion : top
runners : 1st, 2nd, 3rd
outs : 2
batter : Parra
average : .273
bats : left-handed
score : 3-2
goal : escape inning

What operator is selected?
o intentional-walk
o throw-curve-ball
o throw-fast-ball
o None, the system cannot decide.

(r1) If goal is to Escape, I perceive 2 outs, I perceive a runner on 2nd and I
perceive no runner on 1st
then suggest goal intentionally walk batter (Intentional Walk)

(r2) If goal is to Escape I perceive fewer than 2 outs, or I perceive a runner on 1st,
or I perceive no runner on 2nd, or I perceive no runners
then suggest goal to get the batter out via pitching (Pitch)

(r3) If goal is intentional Walk
then suggest intentional-walk operator

(r4) If goal is Pitch and I perceive a new batter who is left/right-handed
then add batter not out, balls 0, strikes 0, bats left/right

(r5) If the goal is Pitch and batter not out
then suggest throw-curve-ball operator

(r6) If the goal is Pitch and batter not out and bats left-handed
then suggest throw-fast-ball operator

(r7) If only one operator has been selected
then send operator to the motor system and add pitch thrown to state
.... ...

```

Figure 195: Exercise System in Action III

What do you think, David? So just like last time we'll set that the goal is to get the batter out via pitching. And we'll add the necessary information to our current working memory, but when we get down here something confusing happens. So we suggest the throw-curve-ball operator because our goal is to pitch and the batter isn't out. But then, we also suggest they throw a fast ball operator because the goal is to pitch, the batter's not out and the batter bats left-handed. So, when we get down to rule seven, which is if only one operator has been selected, it doesn't fire. So our current rules don't actually lead anywhere. We never bottom out. We never figure out what we want to do. Good David. That was the correct answer. So note what has happened? So far we had assumed that the match between the percept and the production rules captioning the personal knowledge was such that, given the percept we always have one rule which will tell us what action to take. It may take some time to get to this rule. Some rules may need to be established. And then only so the rules get established, then only we get the action. But never the less, the system would work. The difficulty now is, there is no rule which tells us exactly what action to take.

21 - Chunking

[Click here to watch the video](#)

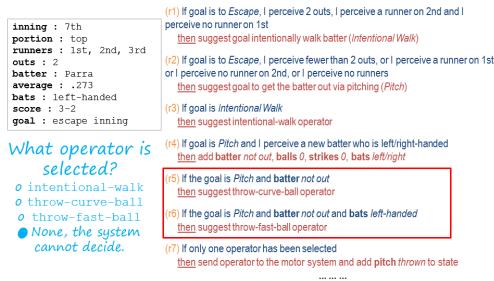


Figure 196: Chunking

So for this situation source cognitive architectures selected not one goal but two. So this SOAR theory this is called an impasse. An impasse occurs when the decision maker cannot make a decision either because not enough knowledge is available or because multiple courses of action there are being selected and the agent cannot decide among them. In this case two actions have been selected and the agent cannot decide between them. Should the pitch throw a curve ball or a fast ball? At this point SOAR will attempt to learn a rule that might break the impasse. If the decision maker has a choice between the fast ball and the curve ball and it cannot decide it, might there be a way of learning a rule that decides between what to throw in a particular situation given the choice of the fast falling curve ball. For this now SOAR will invoke episodic knowledge. Let's see how SOAR does that and how it can help SOAR learn the rule that results in the breaking of the impasse. So imagine that SOAR had episodic knowledge about the previous event, about the previous instance of an event. And this previous instance of an event in another game it was a fifth inning bottom of the fifth inning, if the weather was windy it was the same batter though, Parra, who bats left handed. It was a similar kind of situation and the pitcher threw a fastball and Parra hit a homerun out of it. Now we want to avoid that. The current pitcher wants to avoid it. So given this episodic knowledge about this even set occurred earlier, SOAR has learning mechanism that allows it to encapsulate knowledge from this event into the form of a production rule that can be used as part of the procedural knowledge. And the learned rule is, if two oper-

ators are suggested, and threw a fast ball is one of those operators, and the batter is Parra, then dismiss throw-fast-ball operator. This is the process of learning called chunking. So, chunking is a process, a learning technique that's SOAR uses to learn rules that can break impasse. First note, that chunking is triggered when impasse occurs. In this situation, the impasse is that two rules got activated and there is no way of resolving between them. So the impasse imagery tells the process of chunking, what the goal of chunking is. Find a rule that can break the impasse. SOAR now searches for the episodic memory and finds an event that has some knowledge that may break the impulse. In particular, it looks like a perceptual current situation that we had in previous shot. And compared to the perceptions of previous situations, of the event memory, the episodic memory, and find that any information available of the current batter. If some information is available that tells, SOAR the result of some previous action that also occurs in the current impasse, then SOAR picks that event. So now it tries to encapsulate the result of the previous event, in the form of a rule. In this case, it wants to avoid the result of a homerun, and therefore it says dismiss that particular operator. If it wanted that particular result, it would have said throw that particular operator. We said earlier that in cognitive systems, reasoning, learning and memory are closely connected. Here is an example of that. We're dealing with memory, procedural memory, we're dealing with memory that can deal with procedural knowledge and episodic knowledge. Dealing with reasoning, decision making. We're also dealing with learning, chunking. If you want to learn more about chunking then the reading by Lehman Leodon Rosenblum, and the further readings at the end of this lesson gives lot many more details.

22 - Exercise Chunking

[Click here to watch the video](#)

```

inning : 7th
portion : top
runners : 1st, 2nd, 3rd
outs : 2
batter : Parra
average : .273
bats : left-handed
score : 3-2
goal : escape inning

What operator is selected?
o intentional-walk
o throw-curve-ball
o throw-fast-ball
o None, the system cannot decide.

(r1) If goal is to Escape, I perceive 2 outs, I perceive a runner on 2nd and I perceive no runner on 1st
then suggest goal/intentionally walk batter (Intentional Walk)

(r2) If goal is to Escape, I perceive fewer than 2 outs, or I perceive a runner on 1st, or I perceive no runner on 2nd, or I perceive no runners
then suggest goal to get the batter out via pitching (Pitch)

(r3) If goal is Intentional Walk
then suggest/intentional-walk operator

(r4) If goal is Pitch and I perceive a new batter who is left/right-handed
then add batter not out, balls 0, strikes 0, bats left/right

(r5) If the goal is Pitch and batter not out
then suggest throw-curve-ball operator

(r6) If the goal is Pitch and batter not out and bats left-handed
then suggest throw-fast-ball operator

(r7) If only one operator has been selected
then send operator to the motor system and add pitch thrown to state

(r8) If two operators selected and one has an episode with result homerun
then prefer other operator

```

Figure 197: Exercise Chunking

Let's do one more exercise on the same problem. Note that, I have added one more rule into the procedural knowledge. This is the rule that was the result of chunking. If two operators are suggested, and throw-fast-ball operator suggested, and the batter is Parra. Then dismiss the throw-fast-ball operator. Okay, given these rules and the same situation, what do you think will be the operator that will be selected?

23 - Exercise Chunking

[Click here to watch the video](#)

```

inning : 7th
portion : top
runners : 1st, 2nd, 3rd
outs : 2
batter : Parra
average : .273
bats : left-handed
score : 3-2
goal : escape inning

What operator is selected?
o intentional-walk
 o throw-curve-ball
o throw-fast-ball
o None, the system cannot decide.

(r1) If goal is to Escape, I perceive 2 outs, I perceive a runner on 2nd and I perceive no runner on 1st
then suggest goal/intentionally walk batter (Intentional Walk)

(r2) If goal is to Escape, I perceive fewer than 2 outs, or I perceive a runner on 1st, or I perceive no runner on 2nd, or I perceive no runners
then suggest goal to get the batter out via pitching (Pitch)

(r3) If goal is Intentional Walk
then suggest/intentional-walk operator

(r4) If goal is Pitch and I perceive a new batter who is left/right-handed
then add batter not out, balls 0, strikes 0, bats left/right

(r5) If the goal is Pitch and batter not out
then suggest throw-curve-ball operator

(r6) If the goal is Pitch and batter not out and bats left-handed
then suggest throw-fast-ball operator

(r7) If only one operator has been selected
then send operator to the motor system and add pitch thrown to state

(r8) If two operators selected and one has an episode with result homerun
then prefer other operator

```

Figure 198: Exercise Chunking

So it looks like the entire goal of that chunking process was to help us figure out between these two operators which one we should actually select. We decided that, if a fast ball is suggested, which it was, and if two operators are suggested, which they were, and the batter is Parra, which it is, then to dismiss the throw-fast-ball operator. That means we only have one more operator suggested, throw-curve-ball. So that would be selected.

24 - Fundamentals of Learning

[Click here to watch the video](#)

This is the first time we have come across the topic of learning in this course, so let us examine it a little bit more closely. We are all interested in asking the question, how do agents learn? But this question is not isolated from a series of other questions. What do agents learn? What is the source of their learning? When do they learn? And why do they learn at all? For the purpose of addressing what goal or what task? Now here is the fundamental stance that knowledge based AI takes. It says that we'll start with a theory of reasoning. That will help us address questions like, what to learn, when to learn, why to do learning? And only then will we go to the question, of how to do the learning. So, we reasoning first, and then backwards to learning. This happened in production systems. When the production system reach an impasse, then it said let's learn in order to resolve this impasse from episodic knowledge. So once again, we are trying to build a unified theory of reasoning, memory, and learning where the demands of memory and reasoning constrain the processing of learning.

25 - Assignment Production Systems

[Click here to watch the video](#)

Assignment

How would you use a production system to design an agent that could answer Raven's Progressive Matrices?

Figure 199: Assignment Production Systems

So how would you use a production system to design an agent that can solve Raven's Progressive Matrices. We could think about this kind of at two different levels. At one level we could imagine a production system that's able to address any incoming problem. It has a set of rules for what to look for in a new problem and it knows how to reply when it finds those things. But on the other hand, we can also imagine production rules that are specific to a given problem. When the agent receives a new problem,

it induces some production rules that govern the transformation between certain figures and then transfers that to other rows and columns. So in that way, it's able to use that same kind of production system methodology to answer these problems, even though it doesn't come into the problem with any production rules written in advance. So, inherent in this idea though, is the idea of learning from the problem that it receives. How is this learning going to take place? How is it actually going to write these production rules, based on a new problem? And what's the benefit of doing it this way? What do we get out of actually having these production rules, that are written based on individual problems?

26 - Wrap Up

[Click here to watch the video](#)

To recap...

- Cognitive architectures
- Production systems
- Action selection
- Chunking

to learn a new rule to overcome that impasse. This is the first time we've encountered learning in our course, but learning is actually going to be foundational to everything we talk about from here on. This wraps up the fundamentals unit of our course. Next time we're going to talk about frames, which we actually saw a little bit of today. Frames are going to become a knowledge representation that we'll use throughout the rest of our course.

27 - The Cognitive Connection

[Click here to watch the video](#)

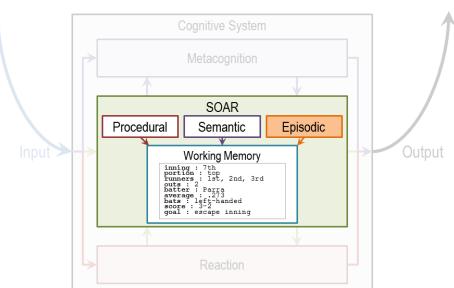


Figure 201: The Cognitive Connection

Figure 200: Wrap Up

So let's wrap up our discussion for today and also wrap up the foundational unit of our course as a whole. We started off today by revisiting this notion of Cognitive architectures that we talked about at the very beginning of the course. We use that to contextualize our discussion of Production systems, specifically those implemented in a framework called SOAR. As we saw, production systems enable action selection. They help us map percepts in the world into actions. Of course, this is only one of the many things that a production system can do. It can really map any kind of antecedent into any kind of consequent. But in our example, we saw how the production system for a pitcher can map percepts from a baseball game into pitch selection. We then talked about impasses and how when a production system hits an impasse, it can use chunking

```

inning : 5th
portion : bottom
game : 131
weather : windy
runners : 1st, 3rd
outs : 1
batter : Pierzynski
average : .283
bats : left-handed
score : 1-4
goal : pitch
pitch : throw-fast-ball
result : homerun
  
```

Figure 202: The Cognitive Connection

```

inning : 5th
portion : bottom
game : 131
weather : windy
runners : 1st, 3rd
outs : 1
batter : Pierzynski
average : .283
bats : left-handed
score : 1-4
goal : pitch
pitch : throw-fast-ball
result : homerun
  
```

(R) If two operators selected and one has an episode with result
homerun
then prefer other operator

“chunking”

Figure 203: The Cognitive Connection

The connection between production systems and human cognition is both powerful and straightforward. In fact, production systems, right from the beginning were proposed as models of human cognition. We can look at it from several perspectives. First, the working memory in production system has a counterpart in human cognition in the form of short-term memory. Short-term memory and human cognition, at least for the verbal part, has a capacity for approximately seven plus or minus two elements. Working memory and production systems plays a similar role. Second, people have connected studies which they have given the same problems, both to humans and to cognitive architectures [affordances]. These problems typically are from closed worlds like arithmetic or algebra. At a consult, there are strong similarities between the behavior of programs like SOAR and the behavior of humans when they address problems in arithmetic and algebra. This, however, does not mean that we already have a very good complete model of human cognition. This is just the beginning. Humans engage in a very large number of problems, not just arithmetic or algebra problems in closed worlds. So there still remain a large number of questions open about how do you build a cognitive architecture that can capture human cognition at large in the open?

28 - Final Quiz

[Click here to watch the video](#)

So we are now at the final quiz for this particular lesson. What did you learn in this lesson?

29 - Final Quiz

[Click here to watch the video](#)

And thank you for doing it.

Summary

This lesson covers the fundamentals of Production Systems which are similar to Expert Systems and based on rules known to domain experts. Production Systems helps map percepts in the world into actions. When the production system reaches an impasse, it uses chunking to learn a new rule to overcome that impasse.

References

1. Winston P., Artificial Intelligence, Chapter 7, Pages 119-137.

Optional Reading:

1. A Gentle Introduction to SOAR; T-Square Resources (GentleIntroductionToSOAR-ExtraReading_Rules_.pdf)
2. Winston Chapter 7, pages 119-137; [Click here](#)
3. Winston Chapter 8, 163-171; [Click here](#)

Exercises

Exercise 1:

In the classical Monkey & Bananas problem, a monkey is faced with the problem of reaching bananas hanging from the ceiling. But a box is available that will enable the monkey to reach the bananas if he climbs on it. Initially the monkey is at location A, bananas at B, and the box at C. The bananas are at height Y, the monkey and the box have height X such that if the monkey climbs on the box, it too will be at height Y.

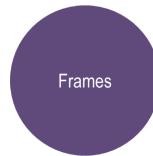
R1: If My-location = Bananas-location AND My-height = Bananas-height, Then Grasp Bananas

R2: If NOT (My-location = Bananas-location) AND My-height = Bananas-height, Then Walk to Bananas-location

R3: If My-location = Box-location AND Box-location = Bananas-location AND My-height ≠ Bananas-height, Then Climb Box to Bananas-height

R4: If My-location = Box-location AND NOT (Box-location = Bananas-location) AND My-height ≠ Bananas-height, Then Push Box to Bananas-location

Lesson 07 - Frames



I am so stereotyped into being this Hollywood girl.
– Kim Kardashian: American actress and model.

If a problem can't be solved within the frame it was conceived, the solution lies in reframing the problem.
– Brian McGreevy, Hemlock Grove (novel).

01 - Preview

[Click here to watch the video](#)

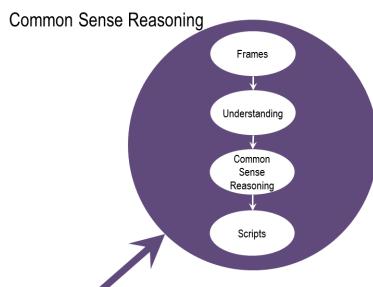


Figure 204: Preview

unit towards building a creative common sense reasoning. They'll also come up in several other topics, such as understanding. We'll begin by talking about the function of frames, what are they useful for, and what makes them so useful. Then we'll talk about several properties of frames. We'll relate the knowledge representation of frames with other knowledge representations, such as semantic networks and production systems. And finally, we will show how frames can be very useful for sense-making. For story understanding.

02 - Exercise Ashok Ate a Frog

[Click here to watch the video](#)

Lesson Preview

- Function of frames
- Properties of frames
- Relationship between frames and previous topics
- Frames for advanced sense-making

Ashok ate a frog.

Is the frog dead or alive?

o dead

o alive

Where is the frog?

o on the table

o on the floor

o in Ashok's stomach

Is Ashok happy or sad?

o happy

o sad

Figure 205: Preview

Figure 206: Exercise Ashok Ate a Frog

Today we'll talk about frames. Frames are a very powerful and very common knowledge representation. Frames are our first step, a basic

We started this unit by saying, that frames are a useful knowledge representation, for en-

abling common sense reasoning. But what is common sense reasoning? You can do it, I can do it. How do we make a machine do it? To illustrate common sense reasoning, let us consider a simple sentence. Ashok ate a frog. All right, you understand the sentence. You understand the meaning of the sentence. But, what is the meaning of the meaning? What did you just understand? Try to answer the questions on the right.

03 - Exercise Ashok Ate a Frog

[Click here to watch the video](#)

- Ashok ate a frog.
- Is the frog dead or alive?
 dead
 alive
- Where is the frog?
 on the table
 on the floor
 in Ashok's stomach
- Is Ashok happy or sad?
 happy
 sad

Figure 207: Exercise Ashok Ate a Frog

I believe I did. So is the first frog dead or alive? I can't imagine you eating a frog and the frog staying alive, so I'm going to say the frog is dead. Where is the frog if you ate it? It's probably in your stomach. And are you happy or sad? I don't know if you like to eat frogs, but I'm going to assume that you do. So I'm going to assume that you are happy. Thank you, David. So David just did common sense reasoning. There was nothing in this input which said whether the frog was dead or alive. There was nothing in the sentence which said Ashok was happy or sad. But David made sensible inferences. I will discuss a lot more about common sense reasoning a little bit later. For now, I want to talk about the knowledge representation called Frames that will allow us to make some inferences of this kind.

04 - How do we make sense of a sentence

[Click here to watch the video](#)

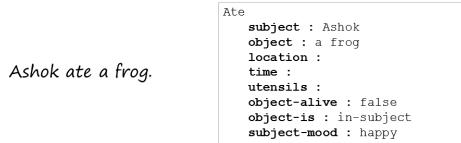


Figure 208: How do we make sense of a sentence

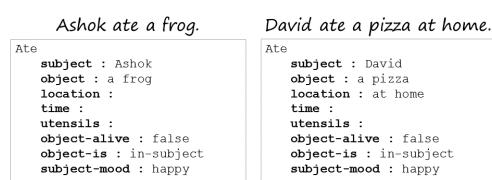


Figure 209: How do we make sense of a sentence

Let us look at the meaning of the sentence, Ashok ate a frog. When I say the sentence, you understand its meaning immediately. But what did you understand? What is the meaning, of the meaning, of the sentence? How can I capture that meaning? How we capture it in a machine? Let us focus for now, on the verb in the sentence, which is ate. We'll associate a frame. With the verb in the sentence. Later on we will see that frames can also be associated with the objects or the nouns in the sentence, but for now we will focus on the verb. Now what is it that we know about the stereotypical action of eating? What happens when people eat, when you and I eat? Usually there is an agent, that does the eating. And that particular agent that corresponds to the subject of the sentence. Usually something is being eaten, that's the object. There is often a location where the eating is done or the time when the eating is being done. Someone might use a utensil to do the eating. You might eat with a fork or a spoon for example. There might be other things that we know about the stereotypical action of eating. For example, what is being eaten typically is not alive. At least not when humans eat it. Now this vertical slot object-is, this

concerns the location of the object. Where is the object after it has been eaten? And you might say well, it's inside the subject's body. What might be the mood of the subject? Well, after people have eaten, typically they are happier. So, here is a list of slots that we associate with the stereotypical action of eating. This is not an exhaustive list, you can add some more. So each of the slots may take some values. We'll call these values fillers. So slots and fillers. Some of the fillers are here by default. Some of the fillers may come from. Parsing the sentence. So, we know that in this particular sentence, the subject is Ashok and the object is a frog. Okay so, frame then, is a knowledge structure. Note the word structure. There are, a number of things happening in this knowledge representation. If I may take an analogy with something, with which I'm sure you are familiar. Consider the difference between an atom of knowledge representation, and a molecule of knowledge representation. Some knowledge representations are like atoms, other knowledge representations are like molecules. An atom is a unit by itself, a production rule is like an atom. On the other hand, frames are like molecules, they have a structure. There are a large number of things happening. These molecules could expand or could contract. You can do a lot more with frames, that you can do with a simple production rule. So frame isn't like a knowledge structure, which has slots, and which has fillers that go with it. Some of these fillers, are by default. A frame deals with the stereotypical situation. Consider now a different sentence. Suppose we had the sentence, David ate a pizza at home. Now here, I have filled out what a frame for this particular sentence would look like. The subject is different, the object is different. This time, there is some information about location, in the previous sentence there was no information about location. Let us compare these two frames for another second. Note that these slots. In case of both the frames are exactly the same, because the frame corresponds to the action of eating. The fillers on the other hand are different, at least some of the fillers are different, because these fillers corresponded the various input sentences. The only fillers that are

the same, are those fillers which have to do with default values for particular slots.

05 - Exercise Making sense of a sentence[Click here to watch the video](#)

Figure 210: Exercise Making sense of a sentence

Okay, let us do an exercise. On the left I have shown you a sentence. On the right is a frame for Ate. Please write down the slots for the frame for Ate, as well as the fillers that will go for these slots for this particular input sentence.

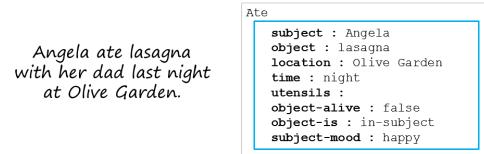
06 - Exercise Making sense of a sentence[Click here to watch the video](#)

Figure 211: Exercise Making sense of a sentence

That's a very good answer, David. First of all, your answer is correct. And in addition, you were able to point out that there is no slot here which is able to take care of the information coming in as input about her dad. Before we worry about what information that Angela had lasagna with her dad. Let us look at some of the properties of frames. So once again there are slots. You can put fillers in the slots. Each filler may come from a range of values. Some of these slots have default fillers. Now you can see how some kind of common sense inferences become possible. Some of them become possible from these

default fillers for these slots. So for example, when we said Ashok ate a frog, and David answered the question whether the frog was dead or alive by saying the frog was dead. Well, we know about it because there is a default value here, the object that has been eaten is not alive, it's false to say it's alive. And the location of the object after it has been eaten is inside the subject's stomach, or inside the subject's body. And so on. We could put here more slots, with more default values for them. Once again, we'll discuss a lot more about common sense reasoning a little bit later in this particular course. For now, we are trying to understand the knowledge and presentation of frames.

07 - Complex Frame Systems

[Click here to watch the video](#)



Figure 212: Complex Frame Systems

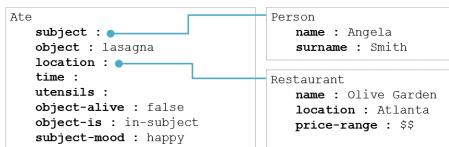


Figure 213: Complex Frame Systems

Now I want to talk about two more ideas. Recall when I had the centers [Ashok ate] a frog, I said lets focus on the verb formula. That was indeed correct, but one can have frames for nouns as well. So one could have a frame for Angela, or for lasagna, or for Olive Garden, or for night. If one had a frame for Angela and lasagna, and

so on, in that case, one could, instead of saying Angela here, point to a frame for Angela. Here is an illustration that shows you that. We may have then information about Angela, quite apart from the sentence about Angela having lasagna with her dad. And that information might be pulled together, with this frame for ate to the pointer. So that now this frame for ate is connected with this frame for Angela. Similarly we may have some information for the restaurant of Olive Garden. We may know for example, that the Olive Garden is located in Atlanta, or that it has a particular price range. So, two ideas. First, not only frames for words, but also for nouns, and second frames can get connected with each other, so that the filler of a slot in one frame, can point to another frame. So Ashok [sp?] it sounds like this kind of representation can be used to make even more advanced inferences. So if I have a frame for Angela, my frame for Angela might have a slot that says food preferences. And under that food preference slot, the filler might be Italian food. And then, when I see this representation, based on that slot and that filler, I can then infer that she's not only happy, she's very happy because she had one of her favorite foods. That's a good point, David. So far we have been talking about sentence level understanding. The sentence that Angela ate lasagna at Olive Garden with her dad, for example. But we can also talk about this cost level understanding, or this cost may contain a series of sentences, one after the other. So the first sentence, for instance, may say something about Angela and her food preferences. And we may construct a frame for Angela. The second sentence may say something about, a restaurant called Olive Garden, which is located in Atlanta. And we may construct a frame for the Olive Garden. When the third sentence comes, and says that Angela had lasagna with her dad at Olive Garden, we construct a frame, and then we can hook up these various frames. And now we beginning to get a disclosed level understanding. Not just a sentence level understanding. So indeed, just like a frame enables us to understand some unit of language. For example, a sentence or a phrase. When we start hooking up these frames together, we can start

understanding larger units of language, for example, a group of sentences of discourse. As we go along, we will see these frames also allow us to pull information from the input sentences, to put in the [Ashok ate] of this slot. So indeed, the ability to hook these frames together, allows for a lot of complex inferences and they will be a major part of common sense reasoning, as we continue with this course.

08 - Properties of Frames

[Click here to watch the video](#)



Figure 214: Properties of Frames

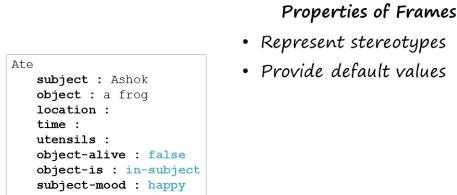


Figure 215: Properties of Frames

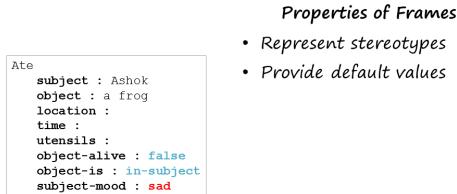


Figure 216: Properties of Frames

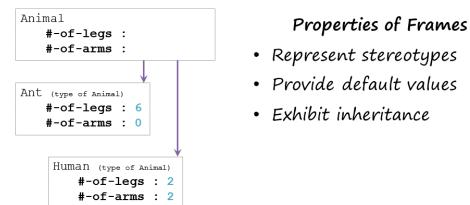


Figure 217: Properties of Frames

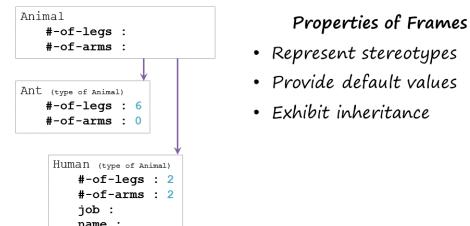


Figure 218: Properties of Frames

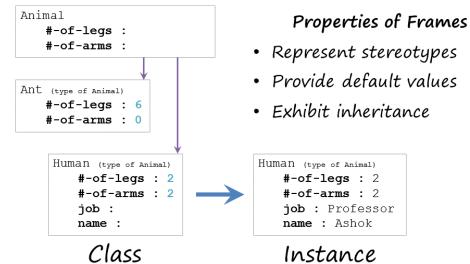


Figure 219: Properties of Frames

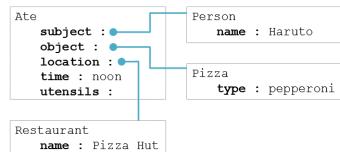
There are three cardinal properties of frames. The first property is that frames represent stereotypes. Now we all know about stereotypes. We deal with stereotypes all the time. Here's a stereotype for the word eat or ate. And this particular stereotype, the slots are dealing with our stereotypical, my stereotypical notion of what happens when something is eaten. There is a subject and it is a frog. There is a location. There is a time. You may have a different set of stereotypes. In fact, stereotypes often are very culture specific. Second, frames provide default values. So not only do they have these slots which come from other notions of a stereotype of this particular word, but many of these slots

may have values already filled in. As an example, I may already have a default value which says that after the object has been eaten, it is no longer alive. It is inside the subject's stomach, and the subject's mood is now happy. There are our default values. Of course, when you have default, you can also have exceptions to them. As an example, it may be that when Ashok ate a frog, it made him sad because frogs don't suit him very well. Now the exception handling is both very powerful and a problem. It is powerful because I can have stereotypes or default values, and when needed, I can override the default values. But it's also a problem, because you can see what will happen. The more instances that I have, the more times where I'll be overriding some default value. And then I have to go to worry about how to manage all of this exceptional handling. Nevertheless, frames provide a very nice way of capturing both default values and exception handling. The third cardinal property of frames is that they exhibit inheritance. So I can organize this frames in a frame hierarchy. Here is a frame for an animal, and then, that has two subclasses, a frame for an ant and a frame for a human. Note, I'm using the language of classes and subclasses here. Now inheritance works, in that I may have some slot for the class animal, and then, I may specify for the ant more specific values for some of those slots. For example, number of legs is six or the number of arms is zero. But, the important thing is that I inherited these slots from the super-class. Of course, when I specify the subclasses, when I go down this frame hierarchy, I may keep on adding additional slots. So for a human man, we may also add the job and the name. There's classes then we have instances, as an example, Ashock is the name of the person and the job is a professor. And so, this instance is also inheriting all these slots and the slot values from the class. We can also see that when the frames provide default values, that's very similar to a constructor when we're dealing with object oriented programming that supplies some initial values when an object is first instantiated. So it seems like there's actually a very rich connection between classes and frames here. David, that's a

very good point. In fact, there is a history to it. Frames and object-oriented programming came about the same time, the 1960s and the 1970s. And I'm sure they influenced each other in both directions.

09 - Exercise Interpeting a Frame System

[Click here to watch the video](#)



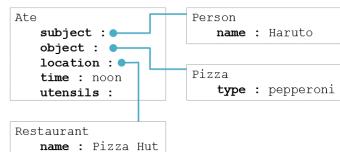
What sentence is expressed by these frames?

Figure 220: Exercise Interpeting a Frame System

Let us do an exercise together. Imagine that there is a set of frames here that is capturing a conceptual knowledge. What sentences is expressed by these frames?

10 - Exercise Interpeting a Frame System

[Click here to watch the video](#)



What sentence is expressed by these frames?

Haruto ate a pepperoni pizza at Pizza Hut at noon.

Figure 221: Exercise Interpeting a Frame System

That's good, David. But here's something interesting to note. It could have been that this was the input sentence and that this frame representation got constructed from this input sentence. So Haruto became the subject and the person and ate became the verb and so on. Alternately, this could have been the frame representation and perhaps the sentence which error gets through language generation from this frame representation. So the frame representation could potentially act as an intermediate

representation for both sentence comprehension and for sentence generation. Of course, there's a lot more to sentence generation and to sentence comprehension than what we have shown so far.

11 - Frames and Semantic Nets

[Click here to watch the video](#)

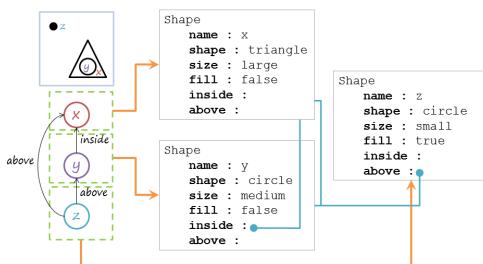


Figure 222: Frames and Semantic Nets

We can also use frames to address the Raven's Matrices problems that we have been talking about all throughout this course. In fact as we do so we'll note another interesting fact, frames in semantic networks are closely related. So let's do this problem. Here is a particular image and here is a semantic network for this particular image that we had come across earlier. I could rewrite this semantic network in the language of frames. But, first of all building a frame for each of these specific objects. I have frame for x, a frame for y and a frame for z. So, here are the frames for the three objects, x, y and z. Let's look at the frame for z in more detail for just a second. So, here are the slots, the name is z, the shape is a circle the size is small and it is filled, you can see it here. We can also capture the relationship between these two objects. So let's consider a relationship example. Here y is inside x, y is inside x. We can capture that through this slot for the object y. Here is the slot for inside, for the object y, and it is pointing to x, indicating that y is inside x. Note again the equivalence between the semantic network and the frame representations. The three objects and the three frames corresponding to three objects. The relationship between the objects and the relationships being captured by these blue lines here between the frames. While we can capture

relationships between frames through lines like this where one frame points to another frame, we could also capture them more directly by actually specifying variables of other frame names. So for example, for the frame y, we might say, inside x which captures the same idea that we were capturing by drawing a line between them. In fact, this is a notation we'll use with the rest of the exercises in this lesson.

12 - Exercise Frames and Semantic Networks

[Click here to watch the video](#)

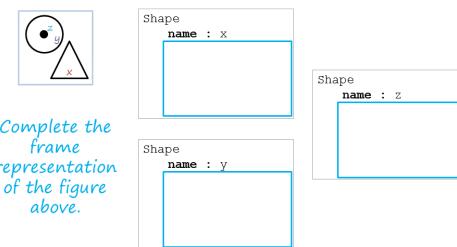


Figure 223: Exercise Frames and Semantic Networks

Let us do an exercise together to make sure that we understand frame representations for images like this. So consider the image shown here on the top left. Can you write down all the slots and the fillers for these three frames?

13 - Exercise Frames and Semantic Networks

[Click here to watch the video](#)

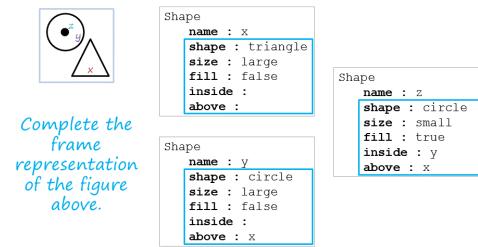


Figure 224: Exercise Frames and Semantic Networks

David, were you able to write down the slots and the fillers. I was. So, for object x, I have that it's a large triangle, that is not filled in. For object y, I have it's a large circle that's not filled

in. And for object z, I have that it's a small circle, that is filled in. As far as the relationships go, in this case we have that z is inside y. Z is inside y. Z is above x, which it is, and y is also above x. Y is above x. Good David, that sounds right to me.

14 - Frames and Production Systems

[Click here to watch the video](#)

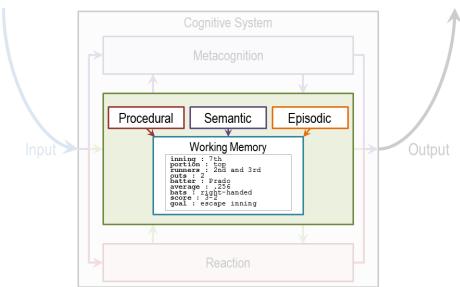


Figure 225: Frames and Production Systems

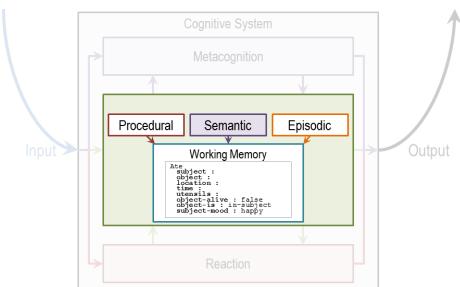


Figure 226: Frames and Production Systems

We have actually come across the notion of frames earlier, when we were talking about production systems. You may recall we had a diagram like this, where we had procedural, semantic and episodic knowledge, and the working memory container structure like this. You can see this is really a frame, here are the slots, here are the values for the slots. We can think of these frames as capturing conceptual knowledge, that is stored in the semantic memory. So let's take an example. Suppose an input is a shark ate a frog. Remember the word ate there, and that verb ate gets returned to working memory, and the entire frame for ate gets pulled out. Once this frame is pulled out of semantic memory, it immediately generates expectations. So we now

know, that ate is likely to have a subject, an object and location, perhaps time utensils and so on. So we can ask ourselves the question, well what will go, under subject here? What will go under object here? And in the sentence a shark ate a frog. This frame tells us, what to look for. As a result of which, the processing is not just bottom up, coming from natural language or the world in general, and going into mind. Also, mind provides knowledge structures like frames, which, structured knowledge representations, which generate expectations and make the processing, partially top down.

15 - Exercise Frames Complex Understanding

[Click here to watch the video](#)

Today, an extremely serious earthquake of magnitude 8.5 hit Lower Slabovia, killing 25 people and causing \$500 million in damage. The President of Lower Slabovia said that the hard-hit area near the Sadie Hawkins fault has been a danger zone for years.

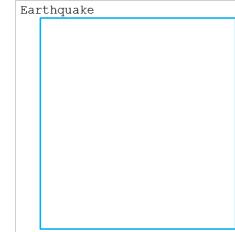


Figure 227: Exercise Frames Complex Understanding

To see both the power and the limitations of the frame knowledge representation, let's do an exercise together. So, please read the story. The story is talking about an earthquake. And then fill out the slots and the fillers that you might think might go with the frame of earthquake.

16 - Exercise Frames Complex Understanding

[Click here to watch the video](#)

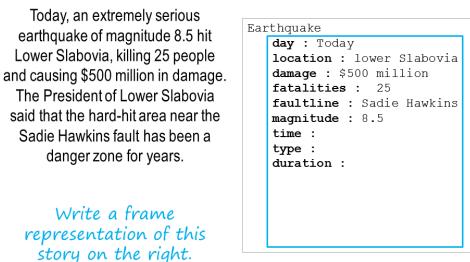


Figure 228: Exercise Frames Complex Understanding

Today, an extremely serious earthquake of magnitude 8.5 hit Lower Slabovia, killing 25 people and causing \$500 million in damage. The President of Lower Slabovia said that the hard-hit area near the Sadie Hawkins fault has been a danger zone for years.

Today, the **President of Lower Slabovia killed 25 proposals totaling \$500 million for research in earthquake prediction**. Our Lower Slabovian correspondent calculates that 8.5 research proposals are rejected for every one approved. There are rumors that the President's science advisor, **Sadie Hawkins**, is at fault.

Figure 229: Exercise Frames Complex Understanding

How do you fill out this frame, David? The way I did it is I started off by writing down what I need to expect about an earthquake. I knew that earthquakes happen on a particular day. They happen at a particular location. They cause a certain amount of damage, a certain number of fatalities. They happen on a certain fault line with certain magnitude. They occur at a certain time. There are different types of earthquakes, and they all have a certain duration to them. I then went to the story to try and find the fillers for the slots I had just written down. So the time was today, the location was Lower Slabovia, and so on. Some slots didn't have fillers in the story, but that's all right. That's good, David. Let us note a couple of things in David's answer. First, the slots are part of David's background knowledge about stereotypical earthquakes. Second, these fillers for the slots are coming from the story. This is a very good example of how both bottom-up processing and top-down processing getting combined. As the story came, David noticed the word earthquake, went into a long-term memory. From a semantic memory he pulled

out the frame for the earthquake which told him what slots to look for, and then he used bottom-up processing to try to figure out what values to put in there. But note the limitation of this combination of top-down and bottom-up processing also. Imagine a different story. Here is a second story. If you read the second story and compare it to the first one, you begin to see that there are lots of subtle and nuanced meanings here. So killed here will happen to 25 proposals, killing here refers to 25 people. Sadie Hawkins here was a science advisor, Sadie Hawkins here was the name of a fault. The frame knowledge representation of itself and by itself doesn't tell us how to pick Sadie Hawkins as a science advisor, and not as a fault. Or whether the killing of 25 proposals, how does it relate with killing of 25 people? We will return to this topic, and we'll talk more about it when we come to understanding and common-sense reasoning in a few weeks. Or if you're interested in reading about this now, you can go ahead and go watch those lessons.

17 - Assignment Frames

[Click here to watch the video](#)

Assignment
How would you use frames to design an agent that could answer Raven's Progressive Matrices?

Figure 230: Assignment Frames

For this assignment, discuss how you'd use frames to represent Raven's Progressive Matrices. At a basic level, what are the slots and fillers associated with different Raven's problems? Where are these frames going to come from? Is the agent going to receive the problem in terms of frames initially, or it going to generate these frames based on its own reasoning? Once it has these frames, what exactly are the reasoning methods it's going to use to solve the problem

based on these frames? We've also talked about frames representing individual figures from the problem. But what about a frame representing the problem, as a whole? What about a frame representing individual shapes within figures? How would representing the problems at that different level of abstraction, help the agents solve the problem more easily. What are frames going to enable us to do, that we couldn't do otherwise?

18 - Wrap Up

[Click here to watch the video](#)

To recap...

- Structure of frames
- Properties of frames
- Frames with other representations
- Story understanding

Figure 231: Wrap Up

So, today we discussed frames, which are one of the knowledge representations that we'll see throughout this course. We started off by talking about the basic structure of frames which involves slots and fillers, and we talked about how similar they are to the variables and values that we see in object-oriented programming. We then talked about the three main properties of frames which are that they represent stereotypes of a certain concept, they provide default values, and they can inherit from one another. We then talked about frames in terms of other concepts we've already covered in this course. We talked about how frames are representationally equivalent to semantic nets, and we've talked about how frames were actually what we were using when we were doing projection systems last lesson. We finally talked about some of the advanced reasoning and story understanding that we can do with frames. Now we're going to move to a topic called learning by recording cases, where we learn from individual cases or individual experiences. But if you're very interested in

frames, you might want to jump forward to our lessons on understanding common sense reasoning and scripts. This will all very heavily leverage what we've learned about frames today

19 - The Cognitive Connection

[Click here to watch the video](#)

Frames are typical character of human cognition. Let us consider three specific ways. First, frames are a structured knowledge representation. We can think of production systems as being atoms of knowledge representation, and frames as being molecules of knowledge representation. A production rule captures a very small amount of information. A frame can capture a large amount of information in organized manner as a packet. Second, frames enable me to construct a theory of cognitive possessing which is not entirely bottom-up, but is partially top-down. I have to see a lot of data from the world. But not all of the cognitive processing is bottom-up. The data results in the retrieval of information from my memory. That information, that knowledge in the form of frames then help to make sense of the data. It has been generating expectations of the world. So then the processing becomes not just bottom-up, but also top-down. Third, frames capture the notion of stereotypes. Stereotypes of situations, stereotypes of events. Now, stereotypes can sometimes lead us to incorrect inferences. Yet you and I have stereotypes of whereas kind of events and situations. So why do we have state effects, because they're cognitively efficient. And why are they cognitively efficient? Because instead of reasoning about the world anew each time, they already have default values associated with them. That's the property of frame. All the default values then, enable me to generate certain number of expectations very rapidly. That's cognitively efficient. Here are three formative connections between frames and human cognition. There are a lot more that we'll get into slowly.

20 - Final Quiz

[Click here to watch the video](#)

Please fill out what you learned in this lesson in this box.

21 - Final Quiz

[Click here to watch the video](#)

Great. Thank you very much.

Summary

Frames represent stereotypes of a certain concept (e.g. situation, event, etc.) and are composed of Slots and Fillers. Frames provide default values for the Slots and can inherit from one another. Frames are representationally equivalent to Semantic Nets. In other words, a frame can capture a large amount of information in an organized manner as a packet. Frames enable us to construct a theory of cognitive processing which is both bottom-up and top-down. The data in the frames generates expectations of the world in a cognitive-efficient manner.

References

1. Winston P., Artificial Intelligence, Chapter 9, Pages 179-190, 197-206.

Optional Reading:

1. Winston Chapter 9, pages 179-182, 202-206;
[Click here](#)

Exercises

Exercise 1:

- (a) Write the frame representation for the following sentence:
Robbie kicked the ball to Suzie.
- (b) Now write the frame representation for:
Suzie kicked it right back at him.
- (c) Invent frames for Robbie, Suzie and Ball.
- (d) Now write the frame representation for the following discourse:
Robbie kicked the ball to Suzie.
Suzie kicked the ball right back at him.

Exercise 2:

- (a) Write the frame representation for the following sentence:
Suzie said she wanted vanilla ice cream.
- (b) Now write the frame representation for:
Robbie said “Me too!”
- (c) Now write the frame representation for the following discourse:
Suzie said she wanted vanilla ice cream.
Robbie said “Me too!”
- (d) Now translate the frame for the second sentence in the discourse into English:

Lesson 08 - Learning by Recording Cases



Science is built upon facts, as a house is built of stones; but an accumulation of facts is no more a science than a heap of stones is a house.
– Henry Poincaré, *Science and Hypothesis*.

01 - Preview

[Click here to watch the video](#)

Lesson Preview

- Learning by recording cases
- Nearest neighbor method
- Cases in the real world
- k-Nearest Neighbor

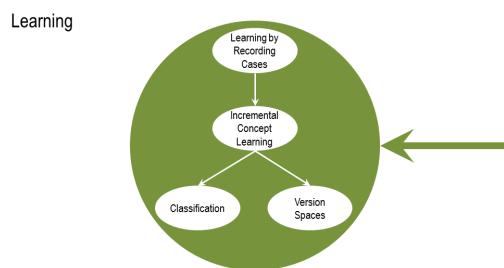


Figure 232: Preview

Figure 234: Preview

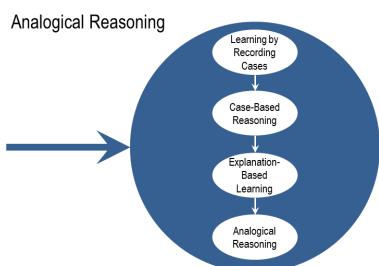


Figure 233: Preview

02 - Exercise Block World I

[Click here to watch the video](#)

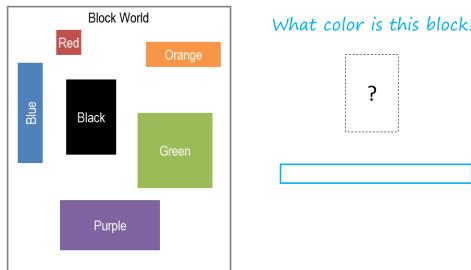


Figure 235: Exercise Block World I

To see how learning, the recording cases might work. Consider a world of blocks. Color blocks, with various shapes and sizes. Six blocks in all. Now let us suppose, that I were to give you a question. So, based on your experiences in this world. What do you think is the color, of this block?

03 - Exercise Block World I

[Click here to watch the video](#)

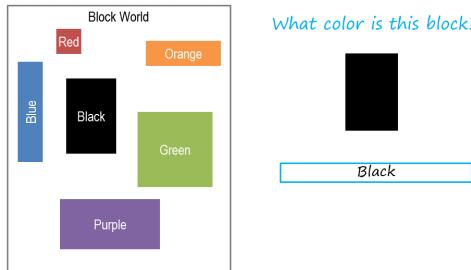


Figure 236: Exercise Block World I

You're right, David. Of the various blocks given here, this block best resembles the black block. And therefore the best guess would be, this is in fact, a black block. This is an example of learning by recording cases, because six cases were recorded in the agent's memory. So when now when a new problem comes along, then the agent gives an answer to that new problem based on the cases that it already had recorded in its memory, it simply sees which case most closely resembles the new situation. And gives the answer to the new situation for that most closely resembling case.

04 - Learning by Recording Cases

[Click here to watch the video](#)

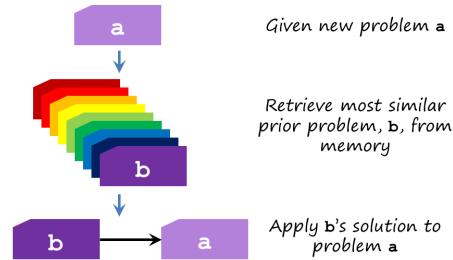


Figure 237: Learning by Recording Cases

This gives a visualization of what is happening in learning by recording cases. There's a new problem a shown by the color here, and memory contains a large number of cases, again represented by the different colors here. So we retrieve the case b that is most similar to the new problem a. In this particular case, we're deciding by color. And whatever was the solution to b, we apply to the situation a. While this visualization is useful, it's still very abstract. Let's think in terms of some practical, everyday examples. So you get up in the morning and you want to go for a run. You put on your sneakers, you have to tie your shoelaces. Well, how do you tie your shoelaces? That's a new problem. But of course, you have tied shoelaces many, many times earlier. So you have a memory of tying shoelaces to different kind of shoes. That's all's to it is cases in your memory. So as you start tying the shoelaces for the shoe today, you simply retrieve the closest matching case and apply it. None of us really thinks very hard in the morning about how exactly to tie shoelaces. If you were to do it, it would take us a very long time. So in learning by recording cases, memory guardedly supplies us with the answer. We don't have to think about it. So another example of this that comes to mind is in programming. Oftentimes in programming, we deal with the same type of problem over and over again. So I might imagine I'm starting out a new program and I'm writing a program in Java. So I'm given a new problem of creating a new program in Java. All I'm going to do is look back in my memory of cases and retrieve another case of starting a new program in Java. And I'm going to apply the solution to that program directly to this one. In this case,

it's going to be something like, when I'm starting a new program in Java, last time, I started with public static void main. I'm going to apply that solution directly to my new problem, and it works. The solution can just be transferred directly to the new problem without any kind of modification. Then later in developing that same program, I might hit, for example, a null pointer error. I'm going to use that null pointer error as a new problem and use it as a probe into my memory. I'm then going to retrieve a case of when I encountered a null pointer error in the most similar program I've worked on, and then that's going to give me a solution that I potentially can apply directly to my current problem. So for example, the solution to the last time I encountered that same error might be to run the program in debug mode and allow it to tell me exactly what variable is null at the time of execution. That's a good example, David. And we could even try to generalize it to medical diagnosis. Imagine that you went to a medical doctor with a set of signs and symptoms. So the doctor is faced with a new problem. What is a diagnosis for your signs and symptoms? The doctor may even have a number of cases recorded in her memory. These are the cases she has encountered during her experience. So the doctor must select the most similar case, the most closely resembling case, which in this case might be b, and say that I will apply to a exactly the same diagnosis that I applied to b. So a case then is an encapsulation of a past experience. And learning by recording cases is a very powerful method that works in a very large number of situations ranging from tying your shoelaces to medical diagnosis.

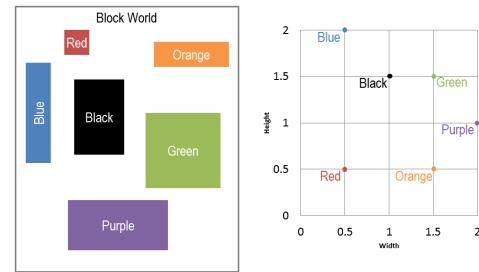


Figure 238: Case Retrieval by Nearest Neighbor

What color is this block?

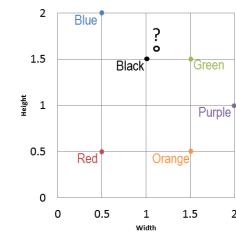


Figure 239: Case Retrieval by Nearest Neighbor

Finding the Nearest Neighbor

Given existing case at (x_c, y_c)
and new problem at (x_n, y_n)

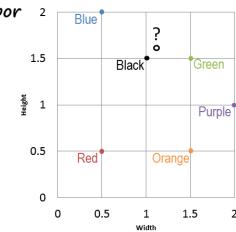
$$d = \sqrt{(y_c - y_n)^2 + (x_c - x_n)^2}$$


Figure 240: Case Retrieval by Nearest Neighbor

Finding the Nearest Neighbor

Given existing case at (x_c, y_c)
and new problem at (x_n, y_n)

$$d = \sqrt{(y_c - y_n)^2 + (x_c - x_n)^2}$$

Block	x_c	y_c	x_n	y_n	d
Blue	0.5	2.0	1.1	1.6	0.72
Red	0.5	0.5	1.1	1.6	1.25
Black	1.0	1.5	1.1	1.6	0.14
Green	1.5	1.5	1.1	1.6	0.41
Orange	1.5	0.5	1.1	1.6	1.17
Purple	2.0	1.0	1.1	1.6	1.08

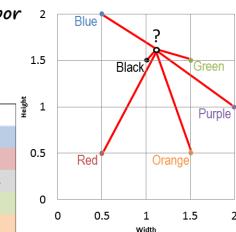


Figure 241: Case Retrieval by Nearest Neighbor

05 - Case Retrieval by Nearest Neighbor
[Click here to watch the video](#)

Let us look at this learning by the recalling cases a little bit more closely. Implicit in our discussion so far has been the notion of most

similar or most closely resembling. But how can we operationalize it? How can we make it more explicit? So once again here is a world of various colored blocks and we can represent these various blocks back to the notion of knowledge for presentation. We can represent our knowledge of these various blocks. In a two dimensional grid, the width of the block and the height of the block. So the blue block may right here, the red block here and so on. So when the new problem comes along we may represent it on the same two dimensional grid. In this particular case the new problem might have been represented in this particular dot. Now given all the cases in the new problem we may calculate the distance between the new problem to each of the previous known cases. Once we have calculated between the new problem and each of the previous cases we can simply select a case which is closest to the new problem. This method is called the nearest neighbor method. Now we need a way of calculating the distance between a problem, in that case. One measure of the distance is called the Euclidean distance. Here is a formula for the Euclidean distance. The Euclidean distance between two points, x of c and y of c , which define the case, and the x of n and the y of n , which define the problem, is given by this formula. Now we can easily calculate the Euclidean distance between each of the cases and new problem, and this table summarizes the distances. Given this table, we can very quickly see that the case of the black block is closest to the new problem and therefore one might give the answer the new block is also black in color. So the nearest neighbor method is one method of finding the most similar case or the most closely resembling case.

What color is this block?

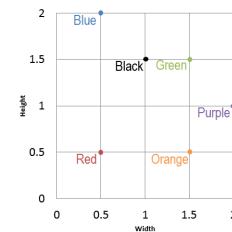
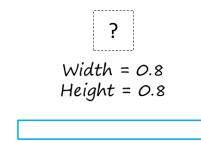


Figure 242: Exercise Retrieval by Nearest Neighbor

Let us do an exercise together. Given the block shown here with the width of 0.8 and the height of 0.8, what do you think is the color of this block?

07 - Exercise Retrieval by Nearest Neighbor

[Click here to watch the video](#)

What color is this block?

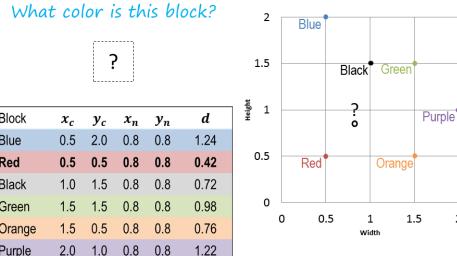
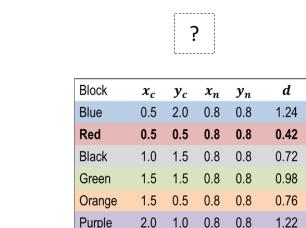


Figure 243: Exercise Retrieval by Nearest Neighbor

Finding the Nearest Neighbor

Given existing case at (x_c, y_c) and new problem at (x_n, y_n)

$$d = \sqrt{(x_c - x_n)^2 + (y_c - y_n)^2}$$

Block	x_c	y_c	x_n	y_n	d
Blue	0.5	2.0	1.1	1.6	0.72
Red	0.5	0.5	1.1	1.6	1.25
Black	1.0	1.5	1.1	1.6	0.14
Green	1.5	1.5	1.1	1.6	0.41
Orange	1.5	0.5	1.1	1.6	1.17
Purple	2.0	1.0	1.1	1.6	1.08

Block	x_c	y_c	x_n	y_n	d
Blue	0.5	2.0	0.8	0.8	1.24
Red	0.5	0.5	0.8	0.8	0.42
Black	1.0	1.5	0.8	0.8	0.72
Green	1.5	1.5	0.8	0.8	0.98
Orange	1.5	0.5	0.8	0.8	0.76
Purple	2.0	1.0	0.8	0.8	1.22

Figure 244: Exercise Retrieval by Nearest Neighbor

So in this problem, we are dealing with a two-dimensional grid, because here, two coordinates, x and y , are enough to represent any one point. In the real world, of course, problems are not

06 - Exercise Retrieval by Nearest Neighbor

[Click here to watch the video](#)

that easy to represent, and one might need a multi-dimensional space in order to be able to represent all the cases in the new problem. Let's examine a problem like that now.

08 - Exercise Recording Cases in Real Life

[Click here to watch the video](#)



Figure 245: Exercise Recording Cases in Real Life

So, here is a map of a small portion of Long Island, New York. Imagine there is an automated car that can navigate the streets of this neighborhood. It comes from the factory with these six cases bootstrapped in it. A, B, C, and so on. For the time being, assume that the car navigates its way in this neighborhood solely by the method of learning where the car in case is. So all it can use is this cases that it knows about. Now, suppose that we have a new problem. The new problem is how to go from Q to this end destination denoted by the arrow. What route is most similar to this new problem?

09 - Exercise Recording Cases in Real Life

[Click here to watch the video](#)



Figure 246: Exercise Recording Cases in Real Life

What do you think, David? So personally I said that D is the most similar case. There are a couple of cases that are closer to the origin, like A and B, but they both go in a very different direction than we're trying to go with Q. Similarly, E ends even closer to Q, but E starts much further away. D looks like it's the one that both starts closest to Q and ends closest to Q. D here is the right answer. Let us think how we can program an air agent to come up with this answer.

10 - Nearest Neighbor for Complex Problems

[Click here to watch the video](#)

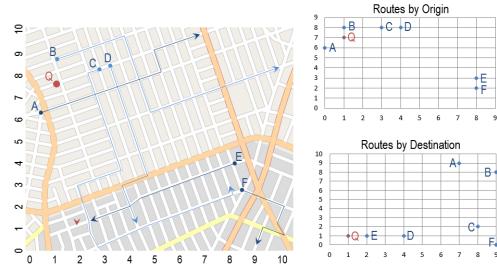


Figure 247: Nearest Neighbor for Complex Problems

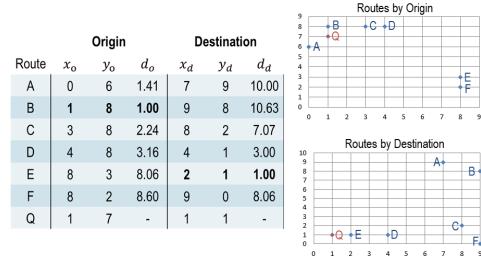


Figure 248: Nearest Neighbor for Complex Problems

Now we can try to calculate the most similar case of the new problem based solely on the origin. The two dimensional grid here tries to represent both all the cases and in your problem. Of course we can also calculate the similarity of the new problem with the old cases based on destination. This two dimensional grid captures the cases and the problem based on the destination. You can compute the [Euclidean] distance from Q in all the cases placed on the origin, shown

here. And you can do the same thing with the destination, shown here. If we focus only on the origin, then the B case seems the closest. If we focus solely on the destination, the E case seems the closest. However, the B case is not very good when we look at the destination. And the E case is not very good when you look at the origin. How then might an AI agent find out which is the best route of all of these choices? How might it decide D is the best route?

11 - Nearest Neighbor in k-Dimensional Space

[Click here to watch the video](#)

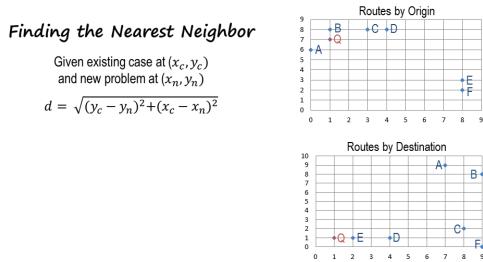


Figure 249: Nearest Neighbor in k-Dimensional Space

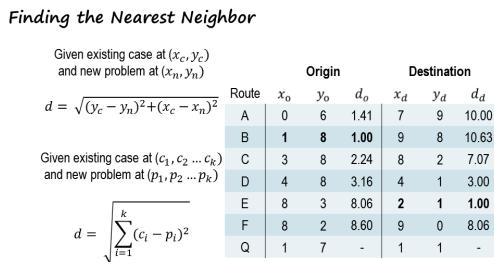


Figure 250: Nearest Neighbor in k-Dimensional Space

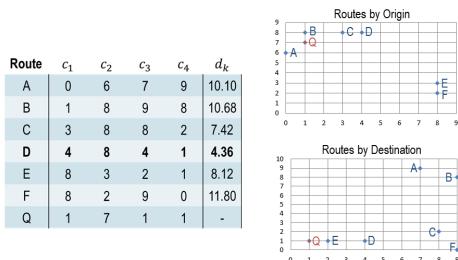


Figure 251: Nearest Neighbor in k-Dimensional Space

Route	c_1	c_2	c_3	c_4	d_k	Route	x_o	y_o	d_o	x_d	y_d	d_d
A	0	6	7	9	10.10	A	0	6	1.41	7	9	10.00
B	1	8	9	8	10.68	B	1	8	1.00	9	8	10.63
C	3	8	8	2	7.42	C	3	8	2.24	8	2	7.07
D	4	8	4	1	4.36	D	4	8	3.16	4	1	3.00
E	8	3	2	1	8.12	E	8	3	8.06	2	1	1.00
F	8	2	9	0	11.80	F	8	2	8.60	9	0	8.06
Q	1	7	1	1	-	Q	1	7	-	1	1	-

Figure 252: Nearest Neighbor in k-Dimensional Space

Earlier we had this formula for calculating the Euclidean distance in two dimensions. Now we can generalize it to many dimensions. So here is a generalization of the previous formula computing nearest neighbor. In this new formula, both the case and the problem are defined in K dimensions. And we'll find the Euclidean distance between them in this K space. So this table summarizes Euclidean distance between the cases and the new problem in this multidimensional space where we are dealing both with the origin as well as the destination are specified by the x and y coordinates. Looking at this table, we can very quickly see that D and not B or E, is the closest case, your most similar case, linear problem Q. This method is called the KNN method where NN stands here for nearest neighbor, K nearest neighbor method. This is a probably method as simple as it is. Of course, it also has limitations. One limitation is that, in the real world, the number of dimensions in which I might want to compute the distance between the new problem and old cases might be very large, a high dimensional low space. In such a situation, deciding which of the stored cases is closest to the new problem may not be as simple as it appears here. A second difficulty with this method is, that even if the new problem isn't very close to an existing case, that does not mean that the existing cases solution can or should be darkly applied to the new problem. So, we need both alternative methods of retrieving cases from memory, and methods for adapting passed cases to fit the requirements of the new problem. That

is called [Case-Based Reasoning] and we will discuss that in the next lesson.

12 - Assignment Learning by Recording Cases

[Click here to watch the video](#)

Assignment

How would you use recording cases to design an agent that could answer Raven's Progressive Matrices?

Figure 253: Assignment Learning by Recording Cases

For this assignment, talk about how you might use notion of recording cases to design an agent that can solve Raven's Progressive Matrices. You might think of cases in a variety of different ways here. For example, each figure in a problem could be a case. Each transformation between figures could be a case. Or more broadly, each problem that your agent has encountered in the past could be a case. As part of this, you'll also need to think about how to evaluate similarity. If you're using figures, how do you evaluate the similarity between two figures in a problem? Or how do you evaluate the similarity between two transformations and a problem? Or more broadly, how do you find what problem that you face in the past, is most similar to the new one you're facing now?

13 - Wrap Up

[Click here to watch the video](#)

To recap...

- Recording and using cases
- Nearest neighbor method
- Cases in real-world problems
- Nearest neighbor in k-dimensional problems

Figure 254: Wrap Up

So today we discussed a learning method called learning by recording cases. In learning by recording cases, we file away individual cases we have encountered in the past in order to use them for future problem solving. We talked about the nearest neighbor method as a way of finding the most similar case to the current problem that we faced in the past. But in the real world, this can often be very difficult. So we talked about using nearest neighbor to find very complex similar cases to our current problem, such as our navigation example. However, there are still a lot of limitations to this method. Oftentimes, just executing a solution we've used in the past doesn't work. And oftentimes, we have to store cases based on qualitative labels instead of numeric labels. These weaknesses will be addressed in our next lesson when we talk about case-based reasoning. There we'll add adaptation and evaluation into our process, and start to be able to use cases in a much more thorough and robust way.

14 - The Cognitive Connection

[Click here to watch the video](#)

Learning by storing cases in memory has a very strong connection to cognition. Cognitive agents like you and I are situated in a world. Our interactions with the world have certain patterns of regularity. The world offers us the same problems again and again. If we think about it, the kinds of problems that you and I deal within a routine everyday basis are the same problems that occurred yesterday and the day before. Tying shoelaces is a good example of that. When we have to tie shoelaces, none of us thinks a lot about how to do it. Memory supplies us with the answer. We don't think as much as we think we do. If you recall we have drawn a cognitive architecture earlier that had three components in it, reasoning, memory, and learning. When we think of intelligence, we typically focus on the reasoning component. We think intelligence has to do with reasoning, with solving problems, with decision making. To some degree, that is true. By learning by recording cases, shifts the balance between the component. It says that, learning is very important and so is memory. We

recall things in memory and then memory supplies us with the answers so that we don't actually have to reason as much as we think we need to.

15 - Final Quiz

[Click here to watch the video](#)

Please write down what all you learned in this lesson, in this box.

16 - Final Quiz

[Click here to watch the video](#)

And thank you for doing it.

Summary

This lesson cover the following topics:

1. Memory is as important as

Learning/Reasoning so that we can fetch the answer to similar cases encountered in the past and avoid having to redo the non-trivial task of learning and reasoning, thereby saving effort.

2. A case is an encapsulation of a past experience that can be applied to a large number of similar situations in future. The similarity metric can be as simple as the Euclidean distance metric or a complex metric involving higher dimensions.

3. kNN method is one method to find the most similar case from memory for a new problem.

4. In some cases, we need to adapt the cases from our memory to fit the requirements of the new problem. In some cases, we also need to store cases based on qualitative labels along with numeric labels to make the comparison applicable for particular situations.

5. Learning by storing cases in memory has a very strong connection to Cognition since human cognition works in a similar manner by recording cases and applying them to new problems in real world by exploiting the patterns of regularity in them.

References

1. Winston P., Artificial Intelligence, Chapter 19.

Optional Reading:

1. Winston Chapter 19; [Click here](#)

2. These Robots Learn How to Cook by Watching YouTube; [Click here](#)

3. The wonderful and terrifying implications of computers that can learn; [Click here](#)

Exercises

None.

Lesson 09 - Case-Based Reasoning



There are three basic approaches to AI: Case-based, rule-based, and Connectionist reasoning.

– Marvin Minsky.

A few observations and much reasoning lead to error; many observations and a little reasoning to truth.

– Alexis Carrel.

01 - Preview

[Click here to watch the video](#)

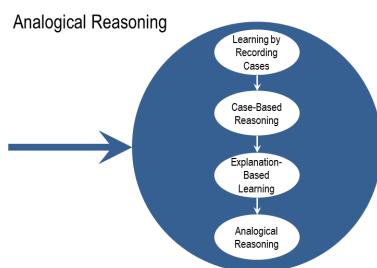


Figure 255: Preview

similar previously encountered problems. Case-based reasoning builds on the previous lesson on learning recording cases. In learning recording cases, the new problem is identical to the previous problem. In case-based reasoning, the new problem is similar to a previously encountered problem. Case-based reasoning typically has several phases, case retrieval, case adaptation, case evaluation, and case storage. We'll also discuss certain advanced processes of case-based reasoning which will include new methods for case retrieval.

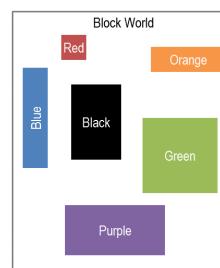
02 - Exercise Return to Block World

[Click here to watch the video](#)

Lesson Preview

- Need for case-based reasoning
- Case adaptation, evaluation, and storage
- Case retrieval revisited
- Advanced case-based reasoning

Figure 256: Preview



What color is this block?

..... ?

Figure 257: Exercise Return to Block World

Today we will talk about Case-Based Reasoning. In Case-based reasoning, the cognitive agent addresses new problems by tweaking solutions to

To illustrate the difference between case-based reasoning and learning by recording cases,

or instance-based learning, let's revisit our micro-world of blocks. Once again, you can see all of these blocks in this micro-world. So you can see the colors, you can see the shapes, and you could even touch them, so you have some idea about their approximate sizes. Now let us suppose I give you a new block. Note that this new block, the size, is very different from the size of any of the other blocks. What color do you think this block will be?

03 - Exercise Return to Block World

[Click here to watch the video](#)

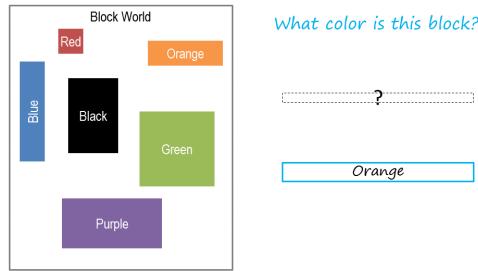


Figure 258: Exercise Return to Block World

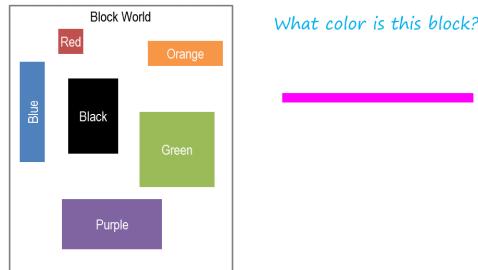


Figure 259: Exercise Return to Block World

And that's the point. The point being that often, the new problem is not identical to the old problem. And when it's not identical, then we have to do some reasoning. We can not just retrieve something from memory, and use the same solution that was used earlier. Case-based reasoning, the phrase case-based reasoning, has two parts to it, case-based, and reasoning. So far we have looked at the case-based part, where we can just extract something from memory and reuse it. Now we can look at the reasoning part. Once you have extracted something off memory, how

can you reason about it, and adapt it, but for the new problem?

04 - Recording Cases to Case-Based Reasoning

[Click here to watch the video](#)



Figure 260: Recording Cases to Case-Based Reasoning

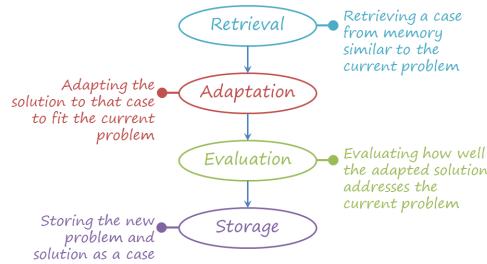


Figure 261: Recording Cases to Case-Based Reasoning

To examine a more realistic problem, let's revisit the problem that we had in our last lesson. Once again, this is a map of a part of Long Island, and the problem is to go from Q to the end location here. So I'll call it Q problem. We'll retrieve from memory the D case, which takes us from this initial location to this collocation. Clearly, this D case is potentially useful for addressing the Q problem. But it is not useful as is. The initial location of the D case is not the same as the initial location of the Q problem. And the end location of the D case is not the same as the end location of the Q problem. So we can start with this D case but we need to adapt it. So, this leads us to the overall process of case-based reasoning. The basic process of case-based reasoning consists of four steps. The first step is retrieval, and we already and considered this

when we were considering learning by recording cases. K nearest neighbor is one way of retrieving cases from memory. Once we have retrieved a case from memory that is delivered to the current problem, we need to adapt it. For example, in the previous problem we had the D case and the Q problem. And we needed to adapt the D case into the Q problem. There are many similar examples. All of us program and all of us, as computer programmers, sometimes use case-based reasoning. We are given a new problem to address, and we often look at the design of a program that we have come across earlier. So there's retrieving a case and they're adapting a particular design of the old program to solve the new problem. Once we have adapted the case to meet the requirements of the new problem, we have a candidate solution for the new problem. With it, the candidate solution is to be evaluated. For example, in the navigation problem, when we have a solution of the Q problem, we can evaluate it but they would actually take us to the end location. We can do a simulation, we can walk through it. As we walk thought it, we will be able to evaluate whether the solution actually succeeds in meeting the requirements of the problem. For the programming problem, once we have a new program that we obtain by adapting the old program, we can actually run the program to see, whether or not it will meet the requirements of the new problem. Let us suppose for a moment that we evaluate a candidate solution and it succeeds. Then, we could encapsulate the new problem and the new solution into a case, and store it back into the case memory, so that case memory is constantly increasing. Notice that this case-based reasoning process unifies memory, reasoning, and learning. There is a case memory that contains a large number of cases and that's how we retrieve cases that are relevant to the current problem. We'll reason when we adapt and evaluate. And we learn when we store the new case back into the case memory.

05 - Assumptions of Case-Based Reasoning
[Click here to watch the video](#)



Assumptions of Case-Based Reasoning

- Patterns exist in the world
- Similar problems have similar solutions

Figure 262: Assumptions of Case-Based Reasoning



Assumptions of Case-Based Reasoning

- Patterns exist in the world
- Similar problems have similar solutions

Figure 263: Assumptions of Case-Based Reasoning

So like any other theory of intelligence, case-based reasoning has some assumptions. The first assumption is that there are patterns to the problems that agents encounter in the world. The same kinds of problem tend to reoccur, again and again. A lot of science is about finding patterns that are crowded in the world. Physics finds patterns that are crowded in the world that are expressed by various kinds of laws, like Newton's second law of motion. Presumably, if you're going to build a theory of intelligence, that theory too will give an account of what kind of patterns exist in the world that mind, that intelligence, must encounter. And case-based reasoning says, one of the common patterns is that the same kinds of problems occur again, and again, and again. So that's the first assumption. The second assumption is that similar problems often have similar solutions. Here is a grid of a part of Long Island. Here is a grid of part of Dallas, Texas. Now if you look at the one in Long Island, you can see that if there are two problems which are very close to each other, they're likely to have very similar solutions. And the same is

true for the grid in Dallas. Similarly, the second assumption is not always valid. Here is an example of it. I'm sure all of you know how to tie your shoelaces. But imagine that you buy a new pair of shoes, and this new pair of shoes have velcro straps. Now the problem's very similar, how to tighten your shoes, but the solution is radically different. So Ashoke, another example of this that comes to mind, for me, is the example of touch screens. Some of the early touch screens could only handle one touch at a time. If you touched it with two fingers at a time, it either wouldn't register, or it'd only register one of the touches. Current touchscreens, on smartphones and tablets that we use today, can handle two or three or four fingers at a time. The problem is very similar. We're still touching the screen and interacting with it with our fingers, but the solution is actually very, very different. It uses a completely different kind of technology, different material for the screen, and a different way of detecting where the screen is being touched. That's a good example, David. So we have at least two examples now where similar problems can have quite different solutions. Nevertheless, this assumption is valid most of the time. Most of the time, two problems that are quite similar will end up having two solutions that are quite similar, as well.

06 - Case Adaptation

[Click here to watch the video](#)

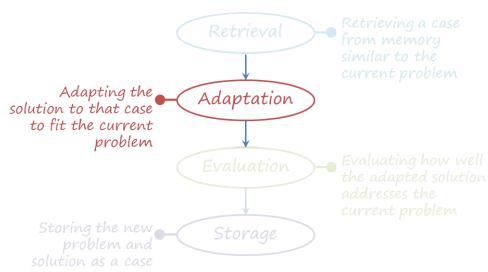


Figure 264: Case Adaptation

Let us look at adaptation a little bit more deeply. So once again here is the process of case-based reasoning. I have kind of blurred the retrieval step. We're going to assume here that

the retrieval of the case has already occurred, perhaps using the KNN method that we discussed last time. Last time we also said two other things. One, that a fundamental conundrum that AR agent faces that the problems they encounter are usually very complex from a computational perspective. And they have only limited calculative resources. So then how can they address computationally complex problems, with limited calculative resources, in near real-time? Seemingly effortlessly. And we said part of the answer might be that memory supplies with an answer. But we are going to make a small amendment to it. Memory supplies with an answer, an almost correct answer so that the adaptation that we have to do is very small, very minor. It's a tweak. As an example of adaptation being mostly tweaking, consider this simple problem, the everyday problem of cooking meals. So you may have a recipe for your favorite kind of meal. And imagine at this time you're going to have your favorite meal, perhaps with different company, or perhaps with a different kind of salad or appetizer. Well in that case you might tweak your recipe for that particular dish. You're not going to change it in a radical way, you're simply making a small change to it. So, Ashoke, another example of this that comes to mind very readily for me, goes back to your programming example from earlier in this lesson. One thing that almost every program I write has to do is input data from a file. But right now, I couldn't write that process from scratch. What I always do is I look at however I did it the previous time, and I'll modify it for the new folder or the new kind of data or the new kind of file I'm reading from. But really I'm just taking the same process and tweaking it for my new problem. That's a good point, David. In fact in the design commonly there's an old cliche, which says that all designers redesign. Design is fundamentally evolutionary, we take all designs and we evolve them slightly and that's how we get a new design. And the same thing is happening in case-based reasoning here. It is saying that often this particular solutions that we come up with are revolutionary in the nature in the sense that they are small tweaks over previous solu-

tions. So the next question becomes, how can we adapt an old case to meet the requirements of new problem? There are potentially several ways of doing it. We will discuss three important ways, perhaps the three most common ways of adapting a case. They are called the model-based method, the recursive case-based method, and the rule-based method.

07 - Case Adaptation by Model of the World

[Click here to watch the video](#)



Figure 265: Case Adaptation by Model of the World



Figure 266: Case Adaptation by Model of the World

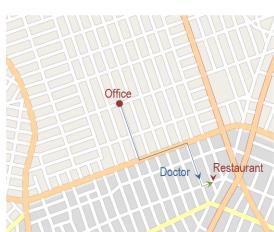


Figure 267: Case Adaptation by Model of the World



Figure 268: Case Adaptation by Model of the World

Let us look at the first, the model base method for adapting our case. Once again we're in this micro world. Let us suppose that we begin from our office, and we need to go to a restaurant. Given the problem of going from the office to the restaurant, let us suppose that we retrieve from memory a case that takes us to a doctor's office, which is quite close to the restaurant but not the same as the restaurant. So one way in which I might be able to allot this case that I have received to address the problem going from the office to the restaurant, is to do a search using this model, this map of the world, which tells me that to go from the doctor's office to the restaurant office, you can take this particular route. So now, I have the earlier case, which I've adapted, using some model of the world. This is an example of using models In order to adapt cases. So in my programming example earlier, instead of having a map of the world, we might have for example an API for interacting with a particular language. I've done input from a file in Python several hundred times, but I've never done it in Java before. I know that the overall process of doing in from Java is going to be very similar to the process of doing it in Python. And I have a model of the way Java works, to know how to actually translate my case of doing in it in Python to doing it in Java. Good example David. It is another one. This one is from design. When we design with this kind of product, let's suppose a VLSI circuit for example, than we not only know something about the configuration of the elements in the design, we also have a model of how that particular configuration is supposed to work. In fact, it might amuse you, David, that

about 25 years back in the 80s, when I wrote my PhD dissertation, it was one of the first PhD dissertations that integrated model-based spacing and case-based spacing. That was exactly the idea in my PhD dissertation. You used models to be able to adapt, evaluate, and store cases.

08 - Case Adaptation by Recursive Reasoning
[Click here to watch the video](#)



Figure 272: Case Adaptation by Recursive Reasoning



Figure 269: Case Adaptation by Recursive Reasoning

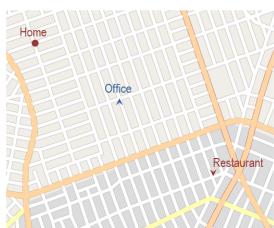


Figure 270: Case Adaptation by Recursive Reasoning

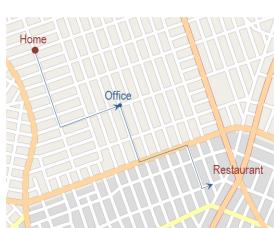


Figure 271: Case Adaptation by Recursive Reasoning

A second method to adapt cases is to use case based reasoning recursively. So last time we considered problem which I had to go from my office to a restaurant, and I found a way of doing that. Now suppose that I have to go from my home to the same restaurant. So I don't yet know how to go from my home to the restaurant, but I know how to go from my home to my office. And this last time, I figured out a way of going from office to the restaurant. So there we have it. Now I have a case retrieved for going from home to office, another case retrieved for going from office to the restaurant, and I have a solution. We're going from home all the way to the restaurant. This is an example of case based reasoning. Your first time you retrieve a case for solving a problem, the case provides a partial solution. So I take the remaining part which was not solved, make it a new problem, and send it back into the case memory. Now the case memory finds me a new case. And I take this new case and I compose it with the previous case to get a full solution. So to return again to our programming example, when I'm designing a program, my file input is usually part of a broader problem of persisting data between instances of the program. And thus, the real problem I'm solving is solving this problem of how to save data when the program isn't running. I can then solve that problem recursively by breaking it down into the first problem of file input and the second problem of file output. I might draw a case for solving file input from one program I've done in the past and a case for solving file output from another program in the past. So I've solved it recursively by breaking it down into sub

problems. David, to build on what you just said, the same kind of thing occurs in design in general. Often when we do design, we get partial solutions for multiple cases. For example, consider the problem of designing a microbot that can swim underwater in a very stealthy manner. This might remind me of a case of a copepod which has large number of appendages and swims in the water at very slow velocity making minimum wake in the water. That's good, I've now solved part of the problem, the part which had to do with moving stealthily underwater under slow speeds. But that now sets up a new goal, how do I achieve stealthy motion underwater at high speeds? And I may come up with a solution from a different case. So a squid, for example, also swims stealthily underwater, but it does so by creating a wake that matches the natural wake of water around it. So here I first used the goal of designing a microbot that can swim underwater stealthily to retrieve a case of the copepod. That provide me with a partial solution. So I set up a new sub-goal to complete the solution. The new sub-goal found a new case, that of the squid, which gave me the rest of the solution. And if I compose the two partial solutions, I get the complete solution. So what Ashok just described is something that we call compound analogy, which is a specific type of adaptation by recursive reasoning. If you're interested in that example, we've provided a paper on it in the course materials for this lesson. So you can read more about the process of adapting those cases to solve that very unique and complex design problem.

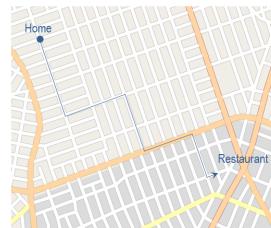


Figure 273: Case Adaptation by Rules

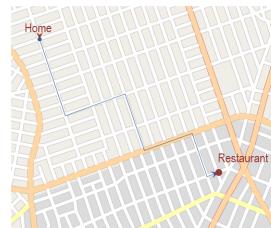


Figure 274: Case Adaptation by Rules

Let us now consider a third method for adapting cases. This method uses heuristics expressed in the form of rules. A heuristic is a rule of thumb. Let's take an example. Imagine that you went to a new city, and you wanted to find out where the downtown was. How could you do that? A simple heuristic is that you just look around and you find where the tallest buildings are. At least in North America, the tallest buildings tend to be in the center, in the downtown of the city. And this heuristic doesn't work all the time. Outside North America, this heuristic sometimes fails. And that's the point. A heuristic is a rule of thumb that works often, but not always. To see how the heuristic method works for adapting cases, consider our problem. Imagine that we're at a restaurant and we need to go back home. Recall that we just found a solution for going from the home to the restaurant. Having found that solution, having evaluated it and executed it, we stored it as a case in memory. So now when we have to go back from the restaurant to the home, we can retrieve that previous case of going from home to the restaurant. Given a new problem and a case, how do we adapt the case to achieve the new problem? In

09 - Case Adaptation by Rules
[Click here to watch the video](#)

this case, we may have a heuristic which says that to go back from where you came originally, you simple have to flip back all the terms. This might give us a solution that is shown here. Note that this heuristic may not always work. It's a rule of thumb. It works often, but of course, we know that we cannot reverse all the turns all the time. So to return to our programming example, we were doing file input, and file input is often a very resource intensive process. So let's say I'm designing a new program and for this new program, efficiency is a much bigger concern. I might have a rule that says when doing file input, it's more efficient to read entire arrays of data at a time instead of just reading one byte at a time. The previous case that I'm adapting might have had file input as 1 byte at a time. But I'm going to use that rule to adapt the case to read arrays of data at a time. So, in that way, that rule has helped me design a file input method that's more efficient. David, to generalize on your answer to design. Designers often use heuristics of the kind that you mentioned. For example, if you want to make an artifact lighter, try a different material. It's a heuristic expressed as a rule.

10 - Case Evaluation

[Click here to watch the video](#)

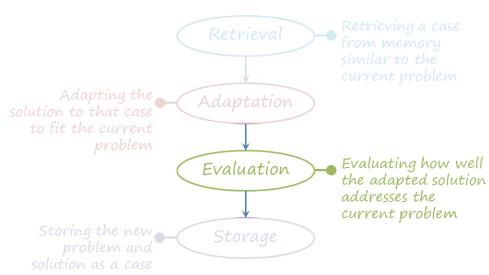


Figure 275: Case Evaluation



Figure 276: Case Evaluation



Figure 277: Case Evaluation

So let us return to our overall process for case based reasoning. We have looked at adaptation a little bit. Let's not look at evaluation. The adaptation step has not given us a candidate solution. The evaluation step is concerned with how to assess the suitability of the candidate solution to the problem at hand. So one method of doing evaluation of a case based solution is through simulation. Consider the problem going from the home to the restaurant. Case based reasoning proposed a candidate solution. In order to find out whether this solution actually works, I can do a simulation. I can even do an execution in the [world]. In this case, we might find out if solution actually works. And we might accept the solution. Now, consider an alternative problem. This time, we have to go from the restaurant to the home, and we decided we would simply flip all the turns. But as we execute the solution, we find out that some of the turns are only one way. Now this particular solution fails. In this particular domain, the cost of executing a solution may be low, therefore we can just go ahead and execute it. In other domains, the cost of execution may in fact be quite high and the best we can do is to first simulate it before we

decide to execute it. So evaluation is built very closely into our programming example. Every time we run a program and see whether or not it worked, we're in fact evaluating whether or not our adaptation successfully solved our new problem. When it didn't, we return to the adaptation phase and we try again or we might return to the retrieval phase, and retrieve another case to use to inform our solution. In design more generally, we can simulate the design, or we can actually prototype a design. Under the matter for evaluating a design could be to share it with other designers and let them critique it. So there are a number of different methods that are possible for evaluation as well.

11 - Case Storage

[Click here to watch the video](#)

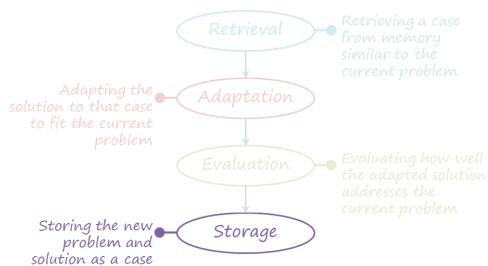


Figure 278: Case Storage

So we just talked about how the evolution step in the case based reasoning process when decided a correct solution in fact meets the requirements of the given problem. Now that we have the new problem and the solution for it, we can encapsulate them as a case, and store them in a case memory. We saw the advantages of this kind of storage earlier, when we went from home to restaurant. We stored that case in memory so that when wanted to go back from restaurant to home. We could retrieve that case and try to adapt it. So case choice is an important way of learning. We are constantly accumulating and assimilating new cases. We talk about two kinds of storage mechanisms. Indexing and discrimination crease.

12 - Case Storage by Index

[Click here to watch the video](#)

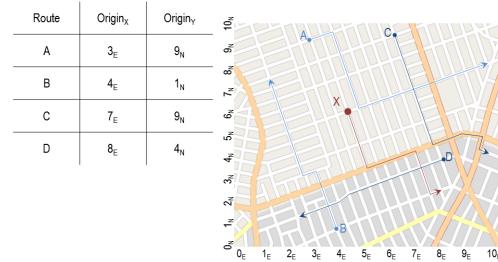


Figure 279: Case Storage by Index

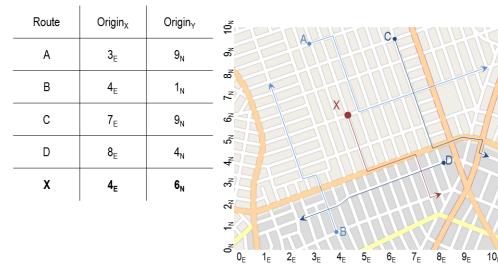


Figure 280: Case Storage by Index

The rest of the notion of indexing, let's go back to our navigation world. Imagine that we already have cases A, B, C and D. We might use a very simple indexing scheme to begin with. We might say well simply index each case with its initial location and the initial location will have a X coordinate and a Y coordinate. So the case A may be indexed by its initial location, which is 3E and 9N and similarly for B, C and D. Now imagine that we have a new case, X of going from the office to the restaurant. Recall that we're indexing cases right now very simply by the XY coordinates of the initial location. So in index case X by the XY coordinates of the initial location here. Let me repeat, this is really a very simple indexical scheme we are using here. As we learned in the lesson last time, we really should be using a more complicated indexical scheme, which takes into account both the initial location and final locations. Nevertheless, this can raise the basic notion of an index. An index is like a tag. At least in principle, we could come up with which index equals key for this particular class of problems? We don't have to limit our social suggesting numerical coordinates of the initial and the goal locations. For

example, in this navigation MicroWorld, the indexes may include whether they are scenic or not scenic, whether their route is fast or not fast. So going back to our programming example we were working with file input and we could have a very rich indexical structure for organizing cases of file input according to various different parameters and variables. For example, I might tag the individual cases of file input according to whether I use Java, Python, C++. I might tag them according to whether there were very fast or very slow and I tagged them according to what kind of file they read in. Did they read text? Did they read XML? Did they read some other kind of file format? Each of those value then becomes a particular way of identifying each individual case, such that when I'm given a new problem, I can find the most similar case by seeing which one matches the most of those variables. That's an important point. We want to use an indexical structure, which allows for effective and efficient retrieval, because we are storing things only, because we want to retrieve them at a later time. In case of design more generally, people have developed indexical structures that have to do with functions, with operating environment, with performance criteria and so on.

13 - Exercise Case Storage by Index I

[Click here to watch the video](#)

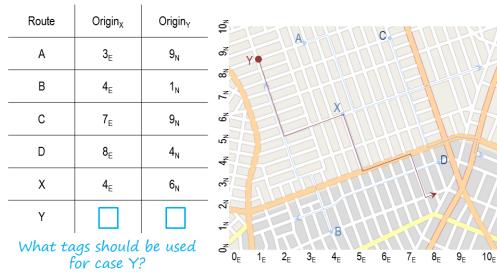


Figure 281: Exercise Case Storage by Index I

But for now, let's go back to where, original navigation micro world. Imagine that we have a nucleus Y. Given our index equals scheme here, of X were coordinates of the initial location, what do you think of the indices of the case Y?

14 - Exercise Case Storage by Index I

[Click here to watch the video](#)

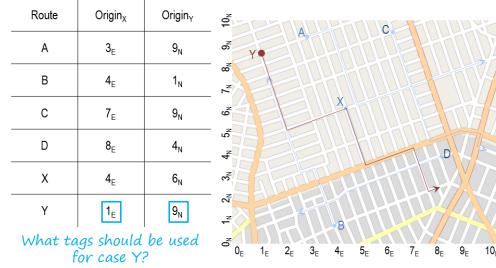


Figure 282: Exercise Case Storage by Index I

Hey Rick, what's your answer? So we can see pretty easily that Y aligns with 1 E, in the horizontal direction, and approximately 9 N in the vertical direction. So, it's 1 E and 9 N. Precisely.

15 - Exercise Case Storage by Index II

[Click here to watch the video](#)

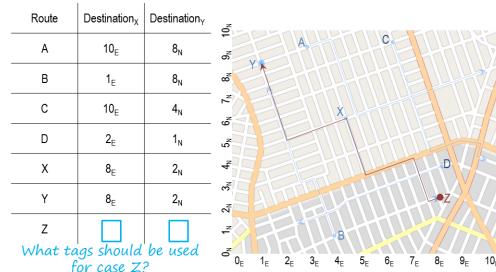


Figure 283: Exercise Case Storage by Index II

Let's consider a different case. Supposing we have a case Z of going back from the restaurant or the home. Let's also suppose that we're change our index equal Kim. Now we are indexing things by the x square coordinates of the destination not the origin. What will be the indices for the case Z?

16 - Exercise Case Storage by Index II

[Click here to watch the video](#)

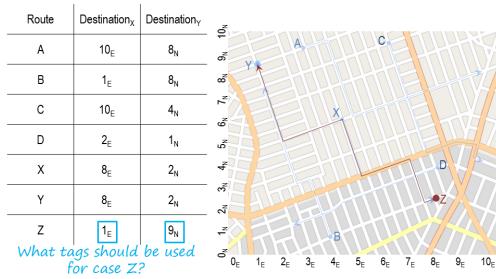


Figure 284: Exercise Case Storage by Index II

That's a good point David. Remember that we're trying to store things, because we want to retrieve things later. And if our storage mechanism is such that it doesn't not allow for efficient retrieval, then it's not a very good storage mechanism. And as you correctly point out, David, as the number of entries increase in this table, and the number of dimensions we are looking at increase also increases. This is going to be coming an inefficient for retrieval. Therefore, let's look at a second method called discrimination trees which provides an alternate way of storing these cases in memory.

17 - Case Storage by Discrimination Tree

[Click here to watch the video](#)

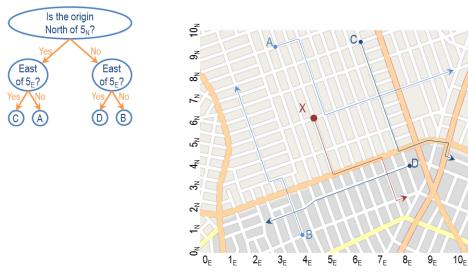


Figure 285: Case Storage by Discrimination Tree



Figure 286: Case Storage by Discrimination Tree

A discrimination tree is a knowledge structure, in which the cases themselves are the leaf nodes of the tree. At the root node, and at all the intimated nodes are questions. The questions of the root node and the intimidated node pertain to the pertain to the indexical structures of the cases. So recall that, we were using the origins of the cases as the index equal structure. Let's stay with that point just a while longer. So now I might have a question that the root node which says is the origin not of 5N? If the answer to that question is yes, then it brings us to this branch. If the answer is no, it takes us to the other branch. At this node I might ask, is the origin east of 5 of E? If yes, it brings us to this branch. If no, it brings us to that branch. In this way we are able to discriminate between C and A, in fact we able to disconnect with C not all of the cases. Similarly for this part of the graph. So now that we have learned, what is the knowledge structure discrimination trees for organization the case memory, let us now look at how will we store a new case. How will we incrementally learn this knowledge structure as new cases are put into the case library? Imagine that there is a new case, X. So we can navigate this tree using X. Is the origin of X North of 5 of A? Yes it is. So we come to this branch. Is the origin of X East of 5 of E? No it is not, so we come to this branch. But now we have a problem. Both A and X, have the same answer no to this question. We must find a way of discriminating between A and X, so we'll add a new question here. Perhaps we can add a new question. Is the origin East of 3 of E? In the case of X, the answer is yes. In the case of A, the answer is no. That's why adding a right node at the right place, we have found a way of discriminating between X and A. This now is a modified discrimination tree. Each time we add a location to memory, the organization of the case of memory changes. This is an example of incremental learning, with the addition of each case some new knowledge structure is learned. We learn more about incremental learning in the next lesson. So going back to our programming example, we were dealing with cases of file input, and we could use the same indexical structure according

to which we organize our cases to now design a discrimination tree. At the very top level I would probably ask, what language is the casing? Is it in Java, C++, Python? Now the discrimination trees don't have to be binary like they are right here. We can have more than two answers coming out. So at the top level, I could have a question of what language is the case in, and the branches could be JAVA, C++, and Python, and so on. I could similarly have questions about, is it an efficient solution, is it for a big problem or a small problem, is it for my personal use or is it for consumer use, and so on until I get down to individual cases that represent different things I might want to consider when I'm doing a new solution. David a point you make about this not being a but a very important one. Let's go back to our original example, where we had a micro world of blocks and the blocks had different colors. So I can ask a question at the root node, what is the color of the block? And have a large number of branches coming out of it corresponding to different colors. Here's an example of a discrimination tree, not a binary print.

18 - Exercise Storage by Discrimin Tree I

[Click here to watch the video](#)



Figure 287: Exercise Storage by Discrimin Tree I

Let us do an exercise. Supposing we're given the case Y, as shown here. And we're given the discrimination tree, shown on the left. Where would you store the case Y in this discrimination tree?

19 - Exercise Storage by Discrimin Tree I

[Click here to watch the video](#)

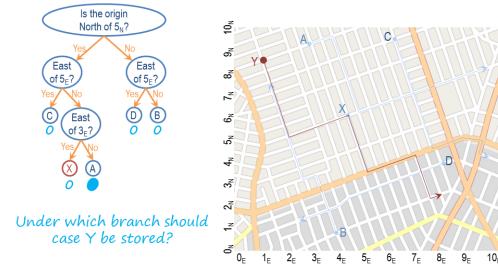


Figure 288: Exercise Storage by Discrimin Tree I

What did you come up with, David? So I started at the root, is the origin North of 5N? It certainly is. Is the origin East of 5E? No, it's not. And is the origin East of 3E? No, it's still not. So I said Y would go down here alongside A. That's good, David, but of course, we must find a way of discriminating between A and Y.

20 - Exercise Storage by Discrimin Tree II

[Click here to watch the video](#)



Figure 289: Exercise Storage by Discrimin Tree II

But know that A and Y were in this same branch. So we now we need to find a way of discriminating between A and Y. How could we do that?

21 - Exercise Storage by Discrimin Tree II

[Click here to watch the video](#)



Figure 290: Exercise Storage by Discriminon Tree II

What did you find, David? So we've got A right here and Y right here. We've got a line that goes through the two that differentiates them that roughly lines up with 2E. So I said 2E. That looks like a good answer to me. Recall that when we were using the table to organize the case memory previously, we were very concerned that this side of the table would grow very large. It will become very difficult to search for a specific case in that table. The potential answer to that. By asking a question we are quickly able to prune away one part of the tree. That makes this search process much more efficient. And that's the point of the discrimination tree. In both organizational scheme, the table and the discrimination tree, we are trying to accommodate and accumulate new cases. But in the case of the discrimination tree, by asking the right questions to the right nodes, we make the search process more efficient. So for those of you familiar with big O notation, you'll notice that the efficiency of searching the case library organized by indices was linear, whereas here it's logarithmic.

22 - Case Retrieval Revisited
[Click here to watch the video](#)

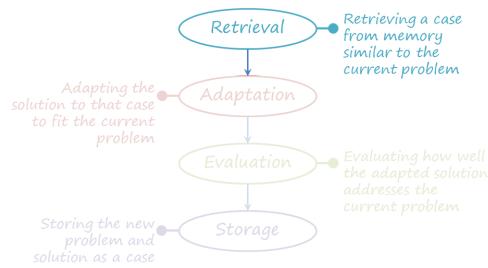


Figure 291: Case Retrieval Revisited

Route	Destination _X	Destination _Y
A	10 _E	8 _N
B	1 _E	8 _N
C	10 _E	4 _N
D	2 _E	1 _N
X	8 _E	2 _N
Y	8 _E	2 _N
Z	1 _E	9 _N

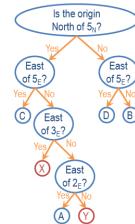


Figure 292: Case Retrieval Revisited

Now that we have considered storage, let's revisit retrieval. We talked about two different ways of organizing the case memory, a tabular way and a discrimination tree. How can we retrieve the case relevant to a given problem? We assume here that the new problem has the same features in its description as the cases stored in the memory. Earlier when we were storing a case in memory, at that time we were navigating this tree to find where in this tree should we store the new case. This time, we'll use the problem to navigate this tree and find out which case is most similar to the problem.

23 - Exercise Retrieval by Index
[Click here to watch the video](#)

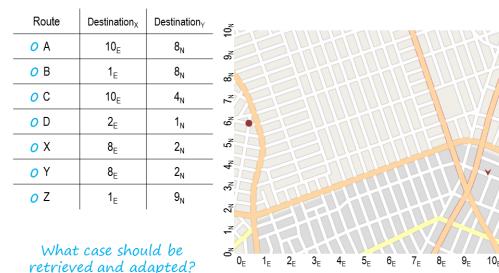


Figure 293: Exercise Retrieval by Index

Let us suppose that the case library is organized in the form of a table as shown here. Let us also suppose that we're given a new problem, how to go from this initial location to this goal location. Which case should be retrieved?

24 - Exercise Retrieval by Index

[Click here to watch the video](#)

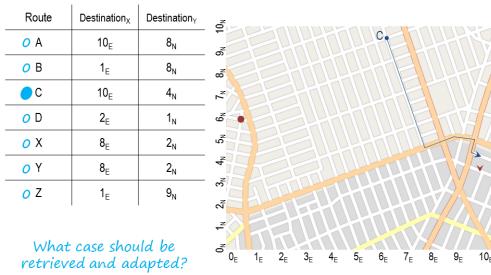


Figure 294: Exercise Retrieval by Index

So what was your answer, David? So comparing the X coordinate of the various cases I found that there's two cases that match the X coordinate of the new problem, A and C. Comparing the Y coordinates, though, A is all the way up here, so I would choose C which matches the X and the Y coordinate exactly. That's right, David.

25 - Exercise Retrieval by Discrimin Tree

[Click here to watch the video](#)



Figure 295: Exercise Retrieval by Discrimin Tree

Let's repeat this exercise, but this time using discrimination tree for organizing the case memory. So here is a discrimination tree, containing the cases currently in the case memory. And here is the, new problem. You could go to

the initial location, to the goal location. Given this problem, what case would be retrieved from this discrimination tree?

26 - Exercise Retrieval by Discrinmin Tree

[Click here to watch the video](#)



Figure 296: Exercise Retrieval by Discrinmin Tree

David what was your answer? So this time we're looking at the origin instead of the destination. We start at the route and ask, is the origin north of 5N? Just barely it is. Is it East of 5E? No, is it east of 3E? No, is it east of 2E? It's really not east of anything. So the case we retrieve is going to be Y. That's right David. Y is the closest matching case to the new problem.

27 - Advanced Case-Based Reasoning

[Click here to watch the video](#)



Figure 297: Advanced Case-Based Reasoning

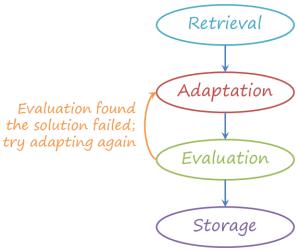


Figure 298: Advanced Case-Based Reasoning

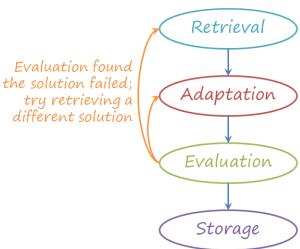


Figure 299: Advanced Case-Based Reasoning

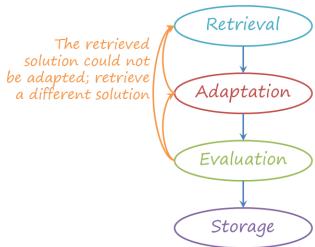


Figure 300: Advanced Case-Based Reasoning

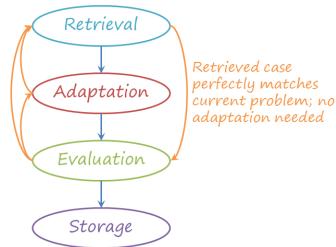


Figure 301: Advanced Case-Based Reasoning

So far we have talked about the very basic process of case based reasoning and we have portrayed as if this process was linear. But of course

the case based reasoning process may not necessarily be linear. As an example, if the evaluation fails, then we might want to adapt that particular case in a different way. As an example of the evaluation of the candidate solution, that adaptation had produced fails. Then, instead of abandoning that particular case, you might want to try to adapt it again. Alternatively, if we try to adapt the same case several times, but we just cannot adapt it, we might want to abandon that case and try to find a different case from the case memory. There is another possibility. Suppose that we retrieve a case from memory. And we try to adapt it but we are unable to adapt it to meet the requirements of the new problem. In that case, you might want to abandon the case and try to do a new one. There is yet another possibility. Let us suppose that we retrieve a case from memory and it exactly matches the new problem. In that case, no adaptation needs to be done and we can jump down to evaluation. In fact this is what happened when we're discussing the KNN method. In this way we can see that there are many ways in which this process need not necessarily be linear. So Ashok, earlier you said that if the evaluation shows that the new solution is good, then we should store it. If the evaluation shows that the new solution is not good, we should try adapting again or we should try retrieving again. But what about evaluation showing that new solution is not good? Should we ever store those? Indeed, sometimes studying failed cases is also very useful. Failed cases can help us anticipate problems. So imagine if you're given a new problem, and you retrieve from your case memory a failed case. That failed case can be very useful because it can help you anticipate the kinds of problems that will occur in solving the new problem. So that reminds me of another example from our file input problem. One thing I've encountered a lot when I'm doing a file input is that if you read too far in the file, then the program will crash and it'll give you an error. It will always give you the same error, and it's a very common problem because different languages do file input slightly differently. So in my mind, I must have cases of the different ways that it's failed in the past, so I can anticipate those and

do it correctly in the future. Failures are great opportunities for learning. When failures occur, we can try to repair the failure by going back from the evaluation step to the adaptation step. Or we can try to recover from the failure by going from the evaluation step all the way to the retrieval step. In addition, we can store these failures in the case memory. When we store them in the case memory, then these failures can help us anticipate failures that might occur with new problems. There's a flip side to this. Just like it is useful to store failed cases, it is not useful to store every successful case. If we stored every successful case, then very soon the case memory will become very, very large, and the retrieval step will become less efficient. This is sometimes called the utility problem. We want to store only those successful cases that in fact help us cover a larger span of problems. This means that, even when a case succeeds, we want to store it only if there is something interesting or noteworthy about that case.

28 - Assignment Case-Based Reasoning

[Click here to watch the video](#)

Assignment

How would you use case-based reasoning to design an agent that could answer Raven's Progressive Matrices?

Figure 302: Assignment Case-Based Reasoning

In this assignment, discuss how you'd use case-based reasoning to develop an agent that can answer Raven's Progressive Matrices. Make sure to describe how this is different from learning by recording cases alone. Where is your adaptation phase? How are you adapting past solutions to the new problem? What is evaluation in this context? How are you evaluating the strength of your answer? Are you going to record the cases that your agent encounters as they're solving the test, or are you going to equip them

with past cases beforehand for them to use to solve new problems?

29 - Wrap Up

[Click here to watch the video](#)

To recap...

- Case adaptation
- Case evaluation
- Case storage
- Case retrieval revisited
- Advanced case-based reasoning

Figure 303: Wrap Up

So today we talked about the broad process of case-based reasoning. Learning by recording cases gave us a method for case retrieval called nearest neighbor method. So we went ahead and jumped into the adaptation phase. Given an old solution to a problem, how do we adapt that old solution to a new problem? We talked about three ways of doing that. We can do it by model of the world, we can do it by rules, or we can do it by recursion. Then once we've adapted that old case, how do we then evaluate how good it was for our problem? Then after we evaluated how good it is we looked at storing it back in our memory. We want to build up a case library of past solutions, so if we've solved a new problem we will now sort that back into our case library. Then based on that we revisited the notion of case retrieval. Based on how our case library is organized, how do we retrieve a prior case that's most similar to our new problem? Now there are a lot of open issues here. For example, should we store failed cases? Should we store failed adaptations? Do we want to store them so we can avoid failing in the future? Should we ever forget cases? Can our case library ever get so big that it's intractable, and we can't really use it efficiently? Should we abstract over cases, so should we use these individual cases to develop a more abstract understanding of a concept, or should we stick the individual cases and adapt them from there? If you're interested in these

questions you can over to our forums and we'll talk about it there. But we'll also be revisiting these questions throughout the rest of the course. Next time we'll talk about incremental concept learning, which takes individual cases and abstracts over them to learn some kind of higher level concepts.

30 - The Cognitive Connection

[Click here to watch the video](#)

Caseless reasoning has a very strong connection with human cognition as well. Analogical reasoning in general is considered to be a core process of cognition. But analogical reasoning depends upon a spectrum of similarity. At one end of this spectrum are problems which are identical to previously encountered problems. In that case, we simply have to retrieve the previous solution and apply it. At the other end of the spectrum, are problems with just semantically very dissimilar from previously encountered problems. We'll discuss those problems later in the class. In the middle of the spectrum are problems. Which are similar, but not identical, to previously encountered parts. So now, we need to retrieve the past solutions, tweak them, and apply them. It is this middle of the spectrum, which is most common in human cognition. Again, going back over cognitive architecture, which had 3 components. Reasoning, learning, and memory. Learning by recording cases shifted the balance from reasoning to learning and memory. Case we can contrast unifies the three of them. It says learning is important because we need to acquire and store experiences. Memory is important because we need to be able to retrieve those experiences when needed. And reasoning is important because we need to be able to tweak those experiences to encounter the needs of new problems.

31 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

32 - Final Quiz

[Click here to watch the video](#)

And thank you for doing it.

Summary

This lesson cover the following topics:

1. In case-based reasoning, the cognitive agent addresses new problems by adapting or tweaking previously encountered solutions to new similar but not identical problems.
2. Case-based reasoning has 4 phases: 1) Case Retrieval, 2) Case Adaptation, 3) Case Evaluation and 4) Case Storage.
3. One method of Case Retrieval is kNN method.
4. 3 common ways of Case Adaptation are: 1) Model-based method, 2) Recursive case-based method, and 3) Rule-based method.
5. Designers often use heuristics for case adaptation. A heuristic is a rule of thumb that works often, but NOT always.
6. Case Evaluation can be performed through Simulation or if the cost is not high then through actual Execution. It can also be done by building a Prototype and testing it or through careful review of the design by experts. Simulation, Prototype or Execution.
7. Case Storage has 2 kinds of mechanisms to organize information for efficient retrieval: 1) Indexing/Tabular method (Linear time complexity) and 2) Discrimination Tree (Logarithmic)
8. Incremental Learning allows the addition of a new case which enables new knowledge structure to be learnt.
9. If the evaluation of a case retrieved fails, then it could be adapted and retried and if the failure continues, then we need to abandon the case. Sometimes, storing failed cases helps us anticipate future problems. Case Adaption is done using model of the world, by using rules or using recursion.
10. We do not need to store all successful cases, but yet need to store noteworthy and representative cases so that we get enough utility from the stored cases and at the same time keep the retrieval process tractable. This is also known as the Utility problem.
11. Case-based reasoning has a very strong connection with human cognition. Case-based

reasoning shifts the balance of importance from Reasoning to both Learning and Memory.

Case-based reasoning unifies all the 3 concepts: Learning (to acquire experiences), Memory (to store and retrieve experiences) and Reasoning (to adapt experiences to similar new problems).

References

1. Kolodner Janet, An Introduction to Case-Based Reasoning.
2. Mantaras & Plaza, Case-Based Reasoning: An Overview.
3. Goel Ashok, Craw Susan, Design, innovation and case-based reasoning.

Optional Reading:

1. Design, innovation, and case-based reasoning; T-Square Resources (Case Based Reasoning 3.pdf)
2. Case-Based Reasoning: An Overview; T-Square Resources (case_based_reasoning_an_overview_1_.pdf)
3. Kolodner, Introduction to Case-Based Reasoning; T-Square Resources (Case Based Reasoning 1.pdf)
4. Case-Based Reasoning: A Love Story; [Click here](#)

Exercises

Exercise 1:

Let us suppose that a top IT company hires you to build a new, personalized, adaptive routing system for navigating urban areas (such as metro Atlanta) by car. Initially, the routing method works as in current systems (such as, say, Mapquest): it has a detailed map of Atlanta and uses a shortest-path algorithm to find a route from a given initial location on the map to the goal location.

However, as people (you, family, friends, perhaps even strangers) use the system, they enter a whole lot of additional information into the system in the form of routing cases: for example, they like some routes that worked very well, they were okay with some routes, they did not care for the routes that ended in failure, etc.

Write/draw a computational architecture, knowledge representations and pseudo-algorithms for a multiple-strategy route planning system that uses the case-based method when an appropriate case is available, and the map-based method otherwise.

Lesson 10 - Incremental Concept Learning



The difficult we do immediately. The impossible takes a little longer.
– US Armed Forces slogan.

Learning and intelligence are intimately related to each other. It is usually agreed that a system capable of learning deserves to be called intelligent; and conversely, a system being considered as intelligent is, among other things, usually expected to be able to learn. Learning always has to do with the self-improvement of future behavior based on past experience.
– Sandip Sen and Gerhard Weiss, *Learning in Multiagent Systems*.

01 - Preview

[Click here to watch the video](#)

Learning

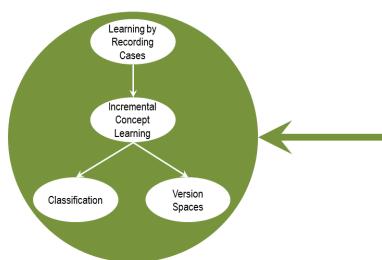


Figure 304: Preview

learning. In case based reasoning, we're storing new examples in memory. Today we'll talk about how we can abstract concepts out of those examples. This is our second major topic of learning. We'll start with the learning goal. Then we'll talk about very near spaces of learning like variabilization, specialization, and generalization. We'll also talk about heuristics for specializing and generalizing concepts as examples come in incrementally.

02 - Exercise Identifying a Foo I

[Click here to watch the video](#)

Lesson Preview

- Purpose of incremental concept learning
- Variabilization
- Specialization
- Generalization
- Heuristics for specialization and generalization

This is a foo.



Figure 305: Preview

Today we'll talk about incremental concept

Figure 306: Exercise Identifying a Foo I

This is not a foo.



Figure 307: Exercise Identifying a Foo I

This is a foo.



Figure 308: Exercise Identifying a Foo I

Yes

No

Is this a foo?



Figure 309: Exercise Identifying a Foo I

Let us try to do a problem together on incremental concept learning. I'm going to give you a series of examples, and we will see what kind of concept one can learn from it. I'll not tell you what the concept really is, for the time being I'm just going to call it foo. Here is the first example. In this first example, there are four bricks. A brick at the bottom, horizontal brick at the bottom, or horizontal brick at the top. And two vertical bricks on the side. Here is a second example. And this time I'll tell you that this particular example is not a positive instance of the concept foo. Once again, we have four bricks, a brick at the bottom, a brick at the top, and two bricks on the side. This time the two bricks

aren't touching each other. Here's a third example of the concept foo. This is a positive example. This is a foo. Again we have four blocks. This time they are two bricks, and instead of having two bricks, vertical bricks, we have two cylinders. They are not touching each other. So I showed you three examples of the concept foo. And I'm sure, you learned some concept definition out of it. Now I'm going to show you another example and ask you, does this example fit your current definition of concept foo, what do you think?

03 - Exercise Identifying a Foo I

[Click here to watch the video](#)

Is this a foo?



Yes

Figure 310: Exercise Identifying a Foo I

And in coming up to this answer David used some background knowledge. The background knowledge that he used was that the bricks that were in the vertical position in the first example, and the cylinders that were in the vertical position in the third example, and the special blocks that are in the vertical position in this example. Are all examples of something called a block. They can be replaced by each other. So instead of having a brick, one could have a cylinder, or some other thing that's vertically placed here. Now, someone else in the class may have a different background knowledge, and he or she may not consider this to be an example of a block, in which case the answer might be no. The point here being that background knowledge is playing an important role in deciding whether or not this is an example of foo.

04 - Exercise Identifying a Foo II

[Click here to watch the video](#)

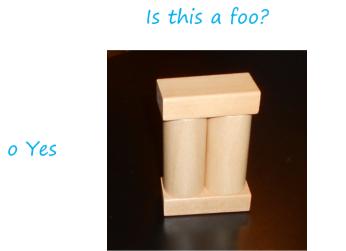


Figure 311: Exercise Identifying a Foo II

Let's try another example. This time, again, there are four blocks. There are two bricks, at the bottom and the top. And the two cylinders, both vertical, but they are touching each other. Is this an example of the concept foo based on what we have learned so far?

05 - Exercise Identifying a Foo II

[Click here to watch the video](#)



Figure 313: Exercise Identifying a Foo III

Lets try one last example in this series. So, in this example, there are four blocks again. There are three bricks at the bottom and two on the side, not touching each other. And there is a wedge this time at the top, not a brick. Is this an example of the concept foo?

07 - Exercise Identifying a Foo III

[Click here to watch the video](#)



Figure 312: Exercise Identifying a Foo II

Once again, David, is using his background knowledge. In his background knowledge he says that the bricks are like the cylinders. The vertical bricks are like the vertical cylinders. So, what holds for the vertical bricks, they must not be touching, also holds for the vertical cylinders. They too must not be touching. Again, someone else may have a different background knowledge and may come up with a different answer.



Figure 314: Exercise Identifying a Foo III

Is this a foo?

**06 - Exercise Identifying a Foo III**

[Click here to watch the video](#)

Figure 315: Exercise Identifying a Foo III



Figure 316: Exercise Identifying a Foo III



Figure 317: Exercise Identifying a Foo III

So to give this answer, David again uses background knowledge. Someone else in the class might say that, well, he does not think that this particular wedge here is the same kind of block as the brick. And therefore this is not an example of foo. So I want to draw a number of lessons from this particular exercise. First, learning is often incremental. We learn from one example at a time. Cognitive agents, human beings, intelligent agents in general, are not always given hundreds or thousands or millions of examples right from the beginning. We get one example at a time. Second, often the examples that we get, are labeled. There is a teacher which tells us this a positive example, or this is a negative example. This is called supervised learning in machine learning literature. Because here there is a teacher which has labeled all the examples for you. Third, the examples can come in a particular order. There are always some positive examples, always some negative examples. The first example, typically is a positive example. Fourth, this is quite different from case based reasoning. In case based reasoning, which we had discussed last time, we had all of these examples, which we stored in their raw form in memory. We'll reuse

them. In this particular case, however, we're abstracting concepts from there. Fifth, the number of examples from which we're extracting concepts is very small. We're not talking here about millions of examples from which we're doing the abstraction. Sixth, when we are trying to abstract concepts from examples, then, what exactly to abstract? What exactly, to learn? What exactly to generalize, becomes a very hard problem. There is a tendency to often overgeneralize, or often, to overspecialize. How does the intelligent agent find out, exactly what is [right kind of] generalization? These are hard questions, and we'll look at some of these questions in just a minute.

08 - Incremental Concept Learning

[Click here to watch the video](#)

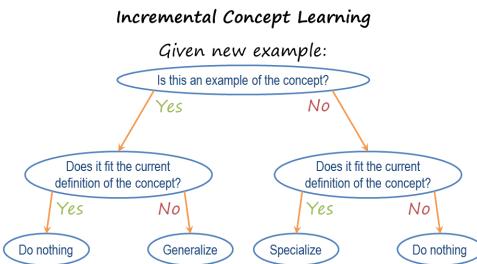


Figure 318: Incremental Concept Learning

Here is the basic algorithm for incremental concept learning and David has created a visual illustration of this algorithm. We're given an example and we're also told whether it's a positive example or a negative example. If this is a positive example, then the algorithm comes to the left branch of this particular tree. And it asks does the current definition of the concept cover this positive example? We want to cover positive examples. If it already covers the positive example, we don't have to do anything. We don't have to devise a current definition of the concept. On the other hand, if the current definition of the concept does not cover the positive example then we must revise it in some way so that it does, so we will generalize it. On the other half of the tree, if this example is not a positive instance of the example, then we can ask ourselves

does the current definition of the concept cover it? If it doesn't cover it, it shouldn't cover it. And if it doesn't cover it, then we don't have to do anything. On the other hand, if the example is a negative instance and if current definition does cover it, then we want to define our current definition to rule it out. So we'll specialize in a current definition. So, oftentimes, we see children committing overgeneralization or overspecialization. So, to take an example of this, we can imagine a child that has a concept of a cat, but the cat has to be black. The child has only ever been around black cats, so part of their definition, part of their concept of the cat is that cats are black. When she goes over to her friend's house, is introduced to her friend's cat, and her friend's cat is orange. Right now, she's told that this is an example of a cat, but it does not fit her current definition of a cat so she needs to generalize her definition that cats can be different colors. Similarly, we can imagine another child that has only ever been exposed to dogs. Thus the child's concept of a dog is that a dog is anything that is furry, has four legs and that we keep as a pet. This child goes over to the same friend's house and is introduced to this orange cat. And right now, that orange cat fits his definition of a dog. It's furry, it has four legs, and they keep it as a pet. But he's told that this cat is not a dog. So he needs to specialize his concept of a dog to exclude this cat. That's good, David. It connects things with our everyday lives.

09 - Variabilization

[Click here to watch the video](#)

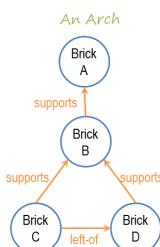


Figure 319: Variabilization

Let us look at the algorithm for incremental concept learning more systematically in more

detail. This time, imagine that there is an AI program, and there is a teacher which is going to teach the AI program about the concept of an arch. So teaching this first example and suppose the teacher gives the example which has four bricks in it. Two vertical bricks that are not touching each other and there is a third brick on top of it and a fourth brick on top of it. To the AI program, the input may look a little bit like this, there are four bricks, A, B, C and D. And there are some relationships between these four blocks. So brick C is on left of brick D. Brick C supports brick B. Brick D supports brick B as well, and brick B supports brick A. This then is the input. What may the error program learn from this one, single example? Not very much. For this one single example, the AI program can only variabilize. There were these constants here, brick A, brick B, brick C, brick D. Instead, the AI program may be able to variabilize these constants and say, well, brick A is an instance of brick, and therefore, I just have brick here. Brick B is an instance of a brick. Therefore, I'll just have a brick here. So now, I can have any brick in these spaces as long as these relationships hold, it's an example of an arch. Note the first example was the positive example. Now we are going to see a series of positive and negative examples, and each time we see an example, the AI program will either generalize or specialize. If it sees a positive example, then it may generalize, if the positive example is not covered by a current concept definition. If it sees a negative example, it may specialize the current definition of the concept to exclude that negative example.

10 - Generalization to Ignore Features

[Click here to watch the video](#)



Figure 320: Generalization to Ignore Features

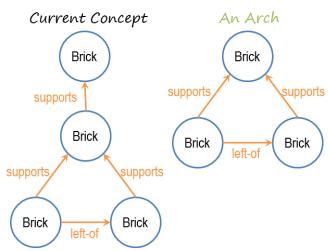


Figure 321: Generalization to Ignore Features

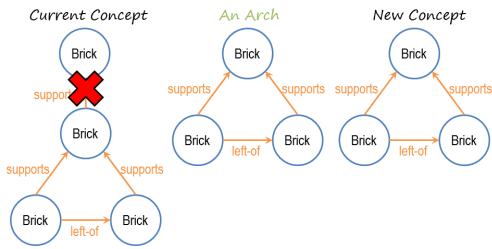


Figure 322: Generalization to Ignore Features

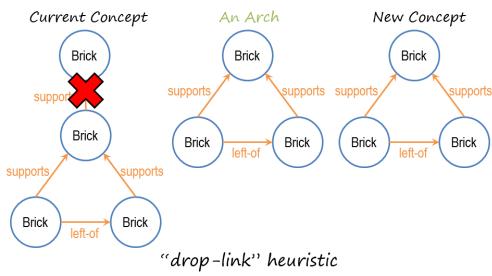


Figure 323: Generalization to Ignore Features

Now suppose that the teacher gives the error program this example as the second example, and the teacher also tells the error program

this is a positive example, so these are labeled examples. Here's a representation of the second example. Again, I have done the variabilization, so the constant here, Brick A, has been replaced by Brick. This was the current concept definition of the AI program for the concept of arch. And here is a new example. How should the AI programmer revise its current concept definition of an arch in order to accommodate this positive example? Because it is a positive example, therefore the AI program should try to generalize. So one good way of generalizing the current concept definition is to drop this link. If the AI program can drop this link, in that case this will be the new current concept definition. Note that this current concept definition covers both the second example, as well as the first example. This is called the drop-link heuristic. It's a heuristic because as we discussed earlier, a heuristic is a rule of thumb. So here is what has happened. When an AI program needs to learn from a very small set of examples, just one example or two examples, then the possible generalizations and specializations, the learning space, is potentially very large. In order to guide the AI program about how to navigate the learning space, we'll develop some heuristics. The first such heuristic is drop a link if you need to generalize in such a way that the new concept can cover both earlier examples. So drop-link heuristic is useful when the structure of the current concept definition and the structure of the new example have a lot of overlap. They overlap almost exactly, except for one link that is extra in the current concept definition. The extra link can be dropped because in the new definition will cover both the previous concept as well as the new example.

11 - Specialization to Require Features

[Click here to watch the video](#)

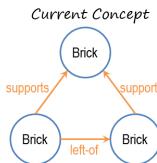
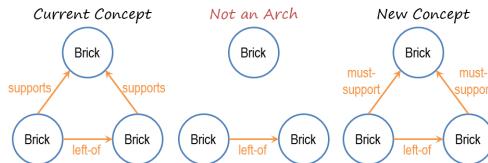


Figure 324: Specialization to Require Features



"require-link" heuristic

Figure 328: Specialization to Require Features



Figure 325: Specialization to Require Features

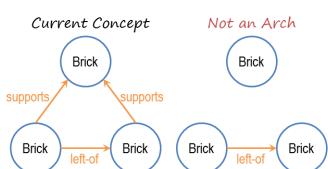


Figure 326: Specialization to Require Features

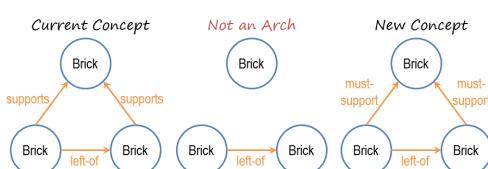


Figure 327: Specialization to Require Features

So here now is the, current concept definition of arch that the AI program has. Now, the teacher shows a new example, here is the new example shown. There are three bricks, but the third brick here, is not on top of the first two. This is the input to the AI program with a third example. And the teacher tells, the AI program that this is not a positive example of an arch. So here is a current concept definition. Here is a representation of the input example, and information that this is a negative instance of the example. What may the AI program learn from it? The AI program must refine its current definition of the arch, in such a way that the, new negative example is ruled out. But how can we do that? One way of doing that is to say, that, we will put extra conditions on these links. These support links must be there. These are not optional. We'll call it, the require-link heuristic. This require-link heuristic says that, if the structure of the presentation of the concept and is structure of the representation of the, negative example have some things in common. But there are also some differences. Then revise the current definition, in such a way that, those things that are not in common become, must be required.

12 - Specialization to Exclude Features
[Click here to watch the video](#)



Figure 329: Specialization to Exclude Features

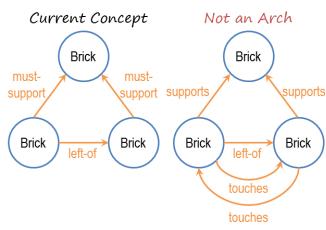


Figure 330: Specialization to Exclude Features

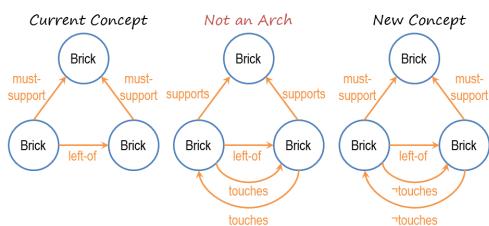


Figure 331: Specialization to Exclude Features

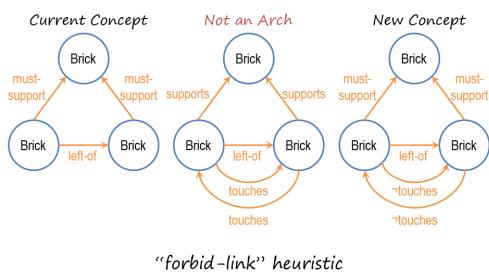


Figure 332: Specialization to Exclude Features

Let us continue this exercise a little bit further. Imagine that the teacher gives this as a third example to the AI program. This time

again you have three bricks, but the two vertical bricks are touching each other. So here is a representation of this input example. Three bricks, the two vertical bricks are supporting this brick at the top, however, the two bricks are touching each other. Recall, this is the current concept definition that the AI program has. It must support links here. And here is the representation of the new example. And the AI program knows that this is a negative example. How might the AI program refine or specialize this particular current definition, so that this negative example is excluded? Well, AI program may say that the current definition can be devised in such a way that for these two bricks, where one is left of the other one, these two bricks cannot touch each other. This particular symbol means not. So it is saying that this brick does not touch that one. And we have bi-dashlink links. Because this one cannot touch the other one, and that one cannot touch this one. This is called a forbid-link heuristic. So, here some particular link, in this particular case touches, is being forbidden.

13 - Generalization to Abstract Features

[Click here to watch the video](#)



Figure 333: Generalization to Abstract Features

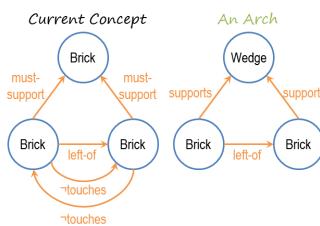


Figure 334: Generalization to Abstract Features

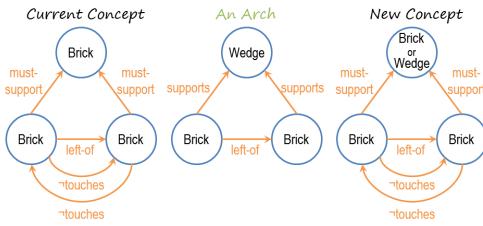


Figure 335: Generalization to Abstract Features

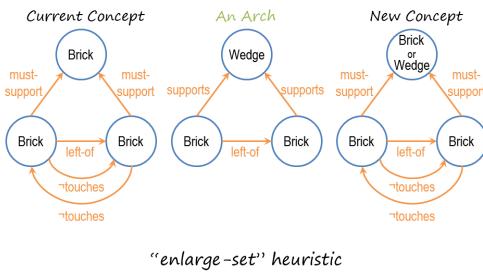


Figure 336: Generalization to Abstract Features

Now let us look at examples that are even more interesting than previously. Recall that earlier we were talking about background knowledge. Let's see what role background knowledge plays more explicitly. So imagine this is the fourth example that the teacher gives to the AI program, and this is a positive example. So the AI program may have this as the input representation. There are two bricks, this brick is left of the other brick, there's a wedge on top, the two bricks are supporting the wedge. So now the AI program has this as the current definition, recall the not touches links here, and this is the new example. And this is the positive example. How may the AI program revise its current concept definition to include this positive example? Well the simplest in the AI program I do is to replace this brick here, in the current concept definition by brick or wedge. So that makes sure that a new example is included in the definition of the concept. We'll call this the enlarge-set heuristic. This particular set here, which had only brick here as an element, now has two elements in it; brick or wedge.

14 - Generalization with Background Knowledge

[Click here to watch the video](#)

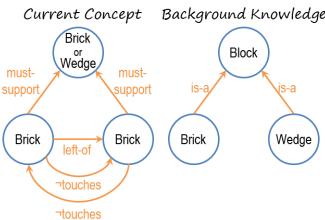


Figure 337: Generalization with Background Knowledge

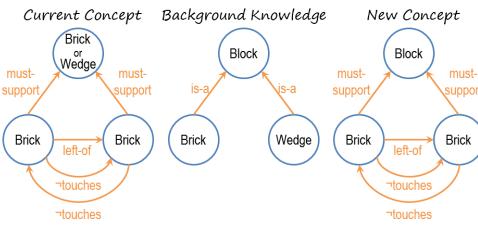


Figure 338: Generalization with Background Knowledge

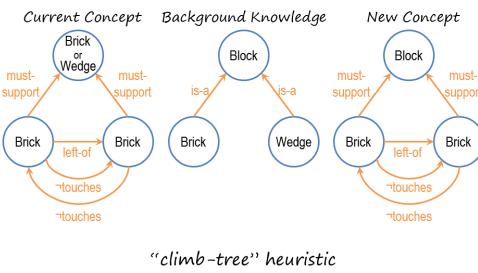


Figure 339: Generalization with Background Knowledge

Now let us suppose that the AI program had background knowledge, which said that brick is a kind of block and wedge is a kind of block as well. In that case, the AI program may use this background knowledge to further revise this current concept definition and replace this brick or wedge by a block, because both bricks and wedges are examples of blocks. So we'll call this a climb-tree

heuristic. Here is a tree of background knowledge, we're climbing the tree. Note another interesting aspect of this. Suppose that there was under the kind of block that was possible, say a cylinder. So now that the AI program has generalized from brick and wedge into a block, and this is the current definition. If a new example comes along and it has a cylinder at the top here, that would be covered by this particular concept. And this AI program would be able to recognize that particular instance which had the cylinder at the top as an example of this concept definition. That is the power of being able to use this background knowledge to generalize. The more we know, the more we can learn. So it sounds like this is actually an example of what I was doing earlier. I had said that because I saw a brick and a cylinder both playing the same role, that they were both examples of block. And therefore, any block could play that role. Then when I saw that funny kind of little arch shape holding up the block on top, I inferred that that would also be an acceptable answer because both brick and cylinders were blocks. And I generalized out to say any block could play that role. Good connection, David.

15 - An Alternative Visualization

[Click here to watch the video](#)

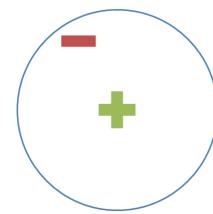


Figure 341: An Alternative Visualization

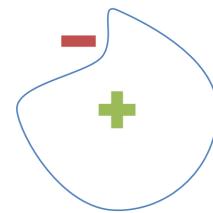


Figure 342: An Alternative Visualization

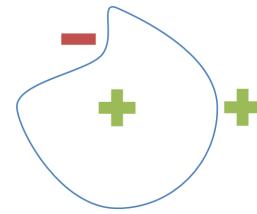


Figure 343: An Alternative Visualization

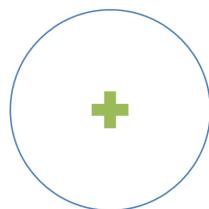


Figure 340: An Alternative Visualization

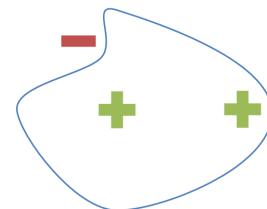


Figure 344: An Alternative Visualization

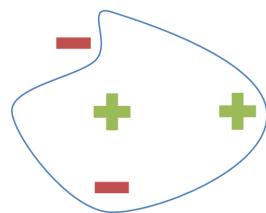


Figure 345: An Alternative Visualization

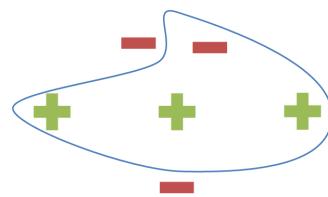


Figure 349: An Alternative Visualization

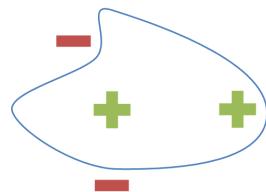


Figure 346: An Alternative Visualization

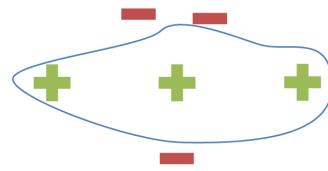


Figure 350: An Alternative Visualization

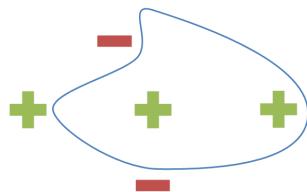


Figure 347: An Alternative Visualization



Figure 348: An Alternative Visualization

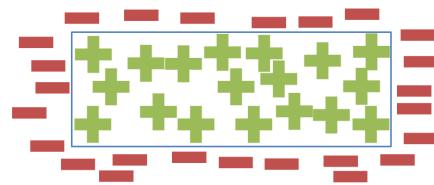


Figure 351: An Alternative Visualization

I hope that algorithm for incremental concept learning makes sense to you. Here is another way of visualization that algorithm. Imagine that an AI agent was given a positive example and the AI agent may come up with a concept definition that covers that positive example. Now let us suppose that the AI agent is given a negative example, and this negative example is covered by the current concept definition. Well in that case the current concept definition must be refined in such a way that the negative example is excluded, while still including the positive example. So you can visualize a new concept definition which includes the positive example, but excludes the negative example. Now let us

suppose that the AI agent is given another positive example, in which case the AI agent must revise its definition of the concept so that the new positive example is also included so that's also covered. So we may revise this concept definition something like this. And we can repeat this exercise many times. Imagine there is a negative example and the current concept definition covers it. Well, we can refine it in such a way that a new negative example is excluded and so on. We can imagine going through several of these iterations of positive and negative examples. Eventually we'll get a concept definition that includes, that covers all the positive examples, and excludes all the negative examples. So again, the problem is the same. Given a small set of positive and negative examples, the number of dimensions in which the algorithm can do generalization and specialization is very large. How do, how do we constrain the learning in this complex learning space? That's where those [heuristics] and background knowledge come in. The [heuristics] guide the algorithm so that it revises the concept definition in an efficient manner and the background knowledge helps in that process.

16 - Heuristics for Concept Learning

[Click here to watch the video](#)

Heuristics for Specializing and Generalizing

require-link: link must be present to be a positive example of the concept

enlarge-set: multiple objects or links may fit one role in the concept

forbid-link: link must be absent to be a positive example of the concept

climb-tree: generalize over multiple objects in the same role based on knowledge

drop-link: link is not necessary to be a positive example of the concept

close-interval: expand range of values to be a positive example of the concept

Figure 352: Heuristics for Concept Learning

Here is a summary of the kind of Heuristics that an AI agent might use in criminal concept learning. You only come across five of them, require-link, forbid-link, drop-link, enlarge-set and climb-tree. Here is another one called close-interval, let's look at it briefly. Let's go to David's example of a child having a dog and suppose that child has only come across dogs that

were very small in size. Now the child comes across a large dog. In that case the child might change the concept definition, expand the range of values that the dog can take, that dog size can take so that the larger dog can be included. So the difference here being that it's values can be continuous like size of a dog and sort of indiscreet as in the other hero sticks

17 - Exercise Re-Identifying a Foo I

[Click here to watch the video](#)

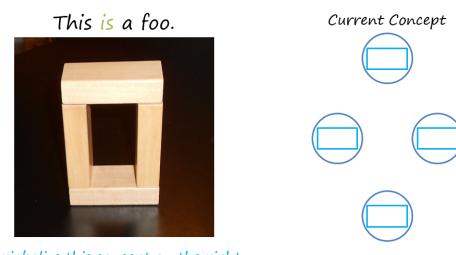


Figure 353: Exercise Re-Identifying a Foo I

Let us do a series of exercises together. This time, the concept that the AI agent is going to learn about, I'll call it foo. Here is a first example the teacher gives to the AI program. All right, because there is only one example the only kind of learning that can occur here is variabilization. What do you think will be the values that can go in these boxes here that will variabilize these four bricks. Initially, they are brick one, brick two, brick three, brick four.

18 - Exercise Re-Identifying a Foo I

[Click here to watch the video](#)

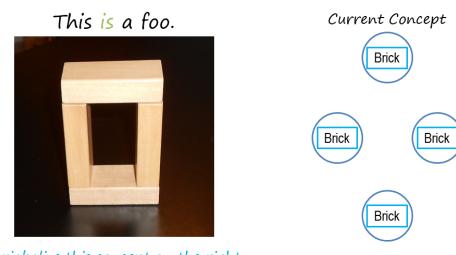


Figure 354: Exercise Re-Identifying a Foo I

So initially we're given labels for these individual bricks. They're brick one, brick two, brick

three, brick four. We're going to abstract over them and say that they're all just instances of bricks. So our four things in this concept are bricks.

19 - Exercise Re-Identifying Foo II

[Click here to watch the video](#)

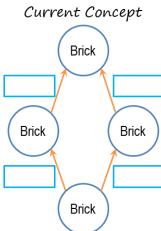


Figure 355: Exercise Re-Identifying Foo II

So how would we reflect the relationship with the concepts on the right?

20 - Exercise Re-Identifying Foo II

[Click here to watch the video](#)

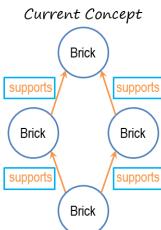


Figure 356: Exercise Re-Identifying Foo II

What do you think, David? So it looks like the brick on the bottom supports the two bricks in the middle. So this brick supports those two bricks. And then those two bricks support the brick on the top. So those two bricks each support that top brick. Note that we've not yet got an example that tells us whether these links are required or not. So that's why these are supports and not must support, as we saw in our concept last time. Now we'll give you some more examples, some positive, some negative. And we'll ask you how the air agent will go about refining its concept definition.

21 - Exercise Re-Identifying Foo III

[Click here to watch the video](#)

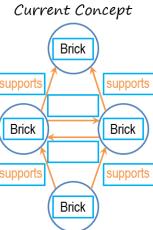


Figure 357: Exercise Re-Identifying Foo III

So let us suppose that the teacher gives this as a second example of the air program. And labels it as a negative instance of the concept, foo. How could the agent refine its current concept definition in order to exclude this negative example?

22 - Exercise Re-Identifying Foo III

[Click here to watch the video](#)

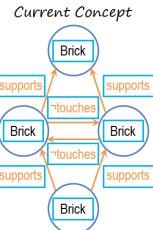


Figure 358: Exercise Re-Identifying Foo III

What do you think, David? So like with our example of the arches, the main thing here is that the bricks here are touching each other. And that's the only real difference between this and our previous example. So we're going to add a new forbidden link here that says the two bricks cannot touch each other. Good job, David.

23 - Exercise Re-Identifying Foo IV

[Click here to watch the video](#)

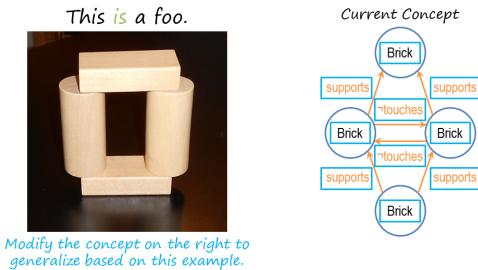


Figure 359: Exercise Re-Identifying Foo IV

Here is the next example. This is a positive example. How would the current definition be refined?

24 - Exercise Re-Identifying Foo IV

[Click here to watch the video](#)

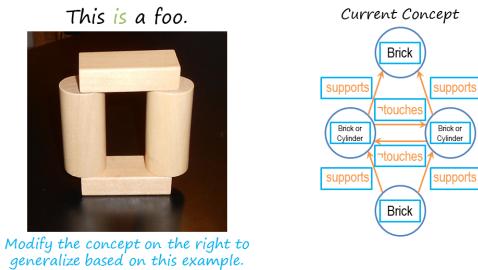


Figure 360: Exercise Re-Identifying Foo IV

That looks good to me, David. And you are right. While humans may have a lot of background knowledge, we have not yet ascribed any background knowledge to the AI agent. So the AI agent might be able to simply say brick or cylinder and nothing more than that

25 - Exercise Re-Identifying Foo V

[Click here to watch the video](#)

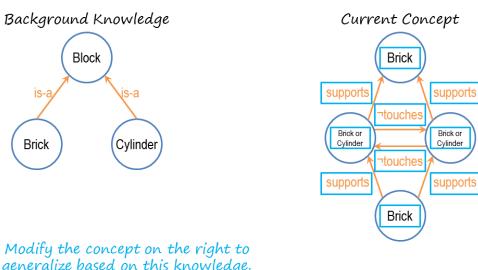


Figure 361: Exercise Re-Identifying Foo V

With this next example, suppose that the AI agent does have some background knowledge which tells it that brick and cylinders are both both sub classes of blocks. In that case, how might the AI program refine this current concept definition?

26 - Exercise Re-Identifying Foo V

[Click here to watch the video](#)

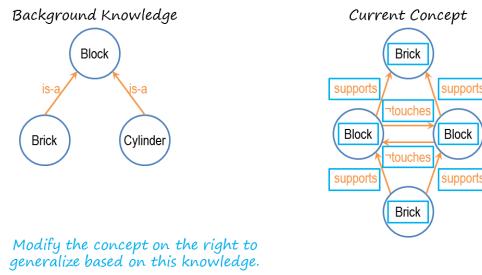


Figure 362: Exercise Re-Identifying Foo V

That's good, David. Notice here the important role that knowledge is playing in learning. Once again, this is why this particular method of incremental concept is part of analogous AI class because we are looking at the critical role that knowledge plays in guiding the learning process. What we learn at the end depends upon what type of knowledge we begin with.

27 - Exercise Re-Identifying Foo VI

[Click here to watch the video](#)

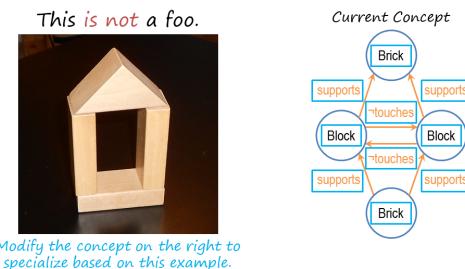


Figure 363: Exercise Re-Identifying Foo VI

Let us consider one last example in this series of examples. Let us suppose the teacher gives this as the negative example of foo. Note, negative example. How do think the AI agent may refine this concept to exclude this negative example?

28 - Exercise Re-Identifying Foo VI

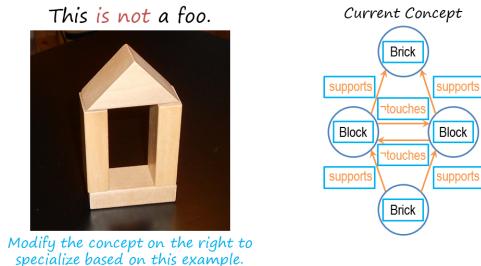
[Click here to watch the video](#)

Figure 364: Exercise Re-Identifying Foo VI

What do you think David? So this is kind of an interesting example. If we recall back in our original tree as to the incremental concept learning process, we saw that there was those do nothing bubbles, that if a negative example does not fit our current definition, we wouldn't actually modify our current definition. Here this doesn't even fit our current concept of a foo. Our current concept of a foo has a brick on top, whereas this is a wedge on top. So this doesn't even fit, so we don't actually need to change our concept, it doesn't fit already. Good, very good. If the example is a negative instance, and the current definition of the concept already excludes it, we don't have to do anything.

29 - Final Concept of a Foo

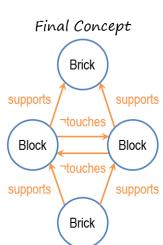
[Click here to watch the video](#)

Figure 365: Final Concept of a Foo

So, given the input series of examples, and the background knowledge, this is the final concept definition for [heuristics] that this particular [heuristics] agent will learn. Notice, that there

are no must support lengths here, because input [heuristics] of examples did not require them. Now it is also, that we did not generalize this bricks into something else, or further generalize these blocks into something else, because there was no background knowledge to do that. So the result of learning here, depends not just on the input examples, but on the background knowledge that the AI agent has. This method of incremental concept learning differs quite a bit from some of these standard algorithms in machine learning. Often in machine learning, the AI agent is given a large number of examples to begin with and the learning begins with those large number of examples where the number of examples could be in thousands or millions or more. When you have a large number of examples to begin with, then one can apply statistical machine learning methods to find patterns of regularity in the input data. But if the number of examples is very small, and if the examples come one at a time, the learning is incremental. Then it becomes harder to apply those statistical methods to detect patterns of the [heuristics] input data. Instead, in that case, the algorithm must make use of its background knowledge to decide what to learn and how to learn it.

30 - Assignment Incremental Concept Learning

[Click here to watch the video](#)Assignment

How would you use incremental concept learning to design an agent that could answer Raven's progressive matrices?

Figure 366: Assignment Incremental Concept Learning

So for this assignment, discuss how you might use incremental concept learning to design an agent that can solve Raven's progressive matrices. What are the concepts that you're trying

to learn in this case? What are you incrementing over? This might depend on the scope of your problem. So are you doing concept learning at the level of the individual problem? Or are you doing it between problems? What concepts are you learning here and how are you actually tweaking them over time? What would a specific concept look like? Or what would a general concept look like? Once you establish these concepts, how will actually help you solve knew problems? Or how are you going to instantiate them or leverage them in new problems that your agent faces?

31 - Wrap Up

[Click here to watch the video](#)

To recap...

- Purpose of incremental concept learning
- Variabilization
- Specialization
- Generalization
- Heuristics for specialization and generalization

Figure 367: Wrap Up

So today we've talked about one particular method for doing what's called incremental concept learning. We started by talking about why we need incremental concept learning. We have instances in the world where we encounter a limited number of examples. We need to discern a concept based on those limited examples. So we start by variabilizing our current understanding to arrive at a more abstract notion of the concept. Then based on new positive or negative examples, we either generalize or specialize our understanding. We talked about a few heuristics for doing this, like the forbid link or require link heuristics. That allow us to develop a better concept based on new examples. Next we'll talk about classification where we leverage the concepts that we developed through incremental concept learning. But we'll also revisit incremental concept learning in some later lessons as well such as through version spaces, explanation

based learning, and learning by correcting mistakes.

32 - The Cognitive Connection

[Click here to watch the video](#)

Incremental concept learning is intimately connected with human cognition. We can adopt two views of learning. In one view of learning, the intelligent agent is given with a large number of examples. The agent's task, then, is to detect patterns of regularity in those examples and learn those patterns of regularity. In the alternative view, the agent is given one example at a time. And the agent has to gradually, incrementally learn concepts out of those examples. Now you and I as agents, as cognitive agents in our daily lives, deal with one example at a time. Rarely is it that we encounter millions or hundreds of thousands of examples given at once. Incremental concept learning is a lot closer than kind of learning that you and I as cognitive agents do.

33 - Final Quiz

[Click here to watch the video](#)

So please write in this box once again what you learned from this particular lesson.

34 - Final Quiz

[Click here to watch the video](#)

Great. Thank you very much.

Summary

Incremental concept learning is intimately connected with human cognition where instead of giving a large number of examples, the agent is given one example at a time and the agent gradually and incrementally learns concepts from those examples.

References

Optional Reading:

1. Winston Chapter 16, pages 349-358; [Click here](#)

Exercises

None.

Lesson 11 - Classification



Science is the systematic classification of experience.
– George Henry Lewes.

01 - Preview

[Click here to watch the video](#)

Learning

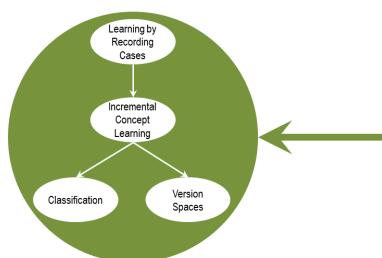


Figure 368: Preview

a hierarchy of concepts. We'll talk about different kinds of concepts, like axiomatic concepts and prototypical concepts. Given a classification hierarchy, we'll talk about multiple processes for doing the classification, including both bottom-up and top-down processes.

02 - Exercise Concept Learning Revisited

[Click here to watch the video](#)

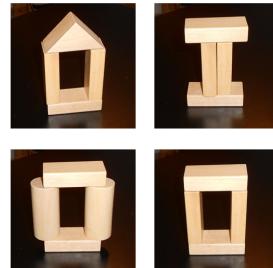


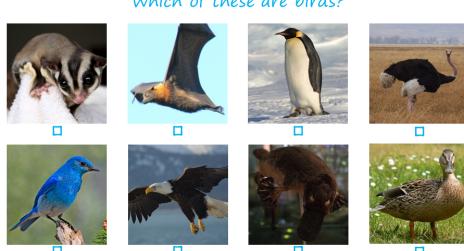
Figure 370: Exercise Concept Learning Revisited

Figure 369: Preview

Today we'll talk about one of the most ubiquitous problems in AI called classification. Classification is mapping sets of percepts in the world into equals classes, so that we can take actions in the world in an efficient manner. We could learn this concept through incremental concept learning. We'll talk about the nature of these equivalence classes and how they can be organized into

Lesson Preview

- Learning concepts
- Equivalence classes
- Concept hierarchies
- Types of concepts
- Bottom-up search



Which of these are birds?

Figure 371: Exercise Concept Learning Revisited

In the previous lesson, we examined some techniques for learning about concepts from examples. But those were simple concepts that we learned from a few examples. Concepts like arg or foo, which was our imaginary, hypothetical concept. The real world concepts can be much more complicated than that. Consider, as an example, consider the eight animals shown here. Each picture shows a very cute animal here. How many of these do you think are birds? Which ones are birds?

03 - Exercise Concept Learning Revisited

[Click here to watch the video](#)

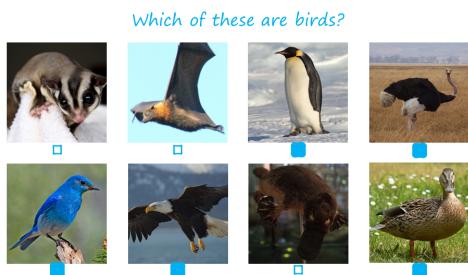


Figure 372: Exercise Concept Learning Revisited

David what did you think and tell us also why you chose the answers that you did. So I said that these five were all birds, each of them have wings, each of them have feathers, and I know that each of them lay eggs. The other three are kind of interesting. This is a sugar glider, and it flies, and we often think of flying as a bird behavior, but it doesn't fly the same way birds do, and it doesn't have the other features that birds have. It has fur instead of feathers and doesn't actually have wings and we also know it doesn't lay eggs. A bat is even more interesting. A bat does have wings, but it doesn't have feathers the way birds do and bats actually give birth to live young. The platypus shown here is an interesting case because it actually lays eggs, it has a bill like a bird does, and it also has flippers the way a duck does. But it's still not a bird because it doesn't have feathers. So I would say that these five are all our birds. Some of the things that we often attribute to birds like flying, the blue bird, and the eagle, and the duck do. Penguins

and ostriches don't actually do, but they're still birds because they meet our other criteria for birds. That was a good answer, David, thank you. Note that David was able to classify these eight animals into birds, which one of these animals belong to the class of birds and which did not belong to it? Notice also that he used some criteria to decide on that. He has some notion it seems of what is a typical bird, what kind of features does a typical bird have. Here's some notion of what are the basic conditions that something must satisfy in order to be considered a bird. And if those conditions are not being satisfied, he would reject them and say those are not birds.

04 - Classifying Birds

[Click here to watch the video](#)



Figure 373: Classifying Birds

So here are four of the animals that David classified as birds. Let us look at the kind of features that he examined in order to classify whether or not an animal was a bird. He may have used several features, some of which he articulated, others that he may not have articulated. So whether an animal has wings? Whether it has feathers? Whether it has a beak? And so on and so forth. One could add more features here if one wished to do so. We do classification all the time. AI agents need to do classification all the time also. Why? Why is classification so ubiquitous?

05 - The Challenge of Classification

[Click here to watch the video](#)

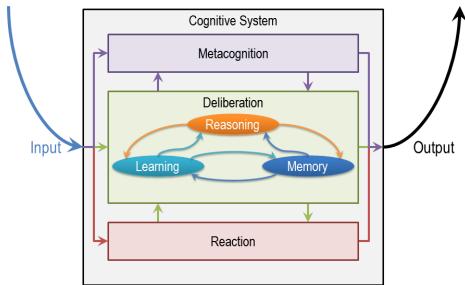


Figure 374: The Challenge of Classification

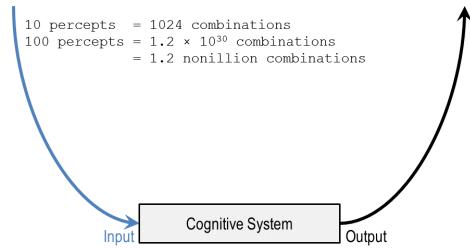


Figure 378: The Challenge of Classification

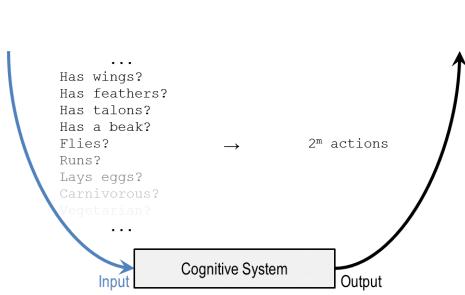


Figure 375: The Challenge of Classification

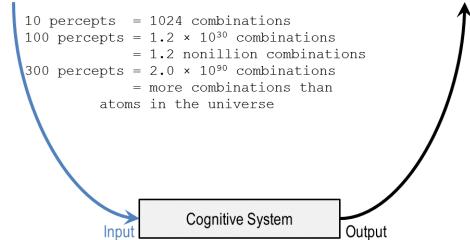


Figure 379: The Challenge of Classification

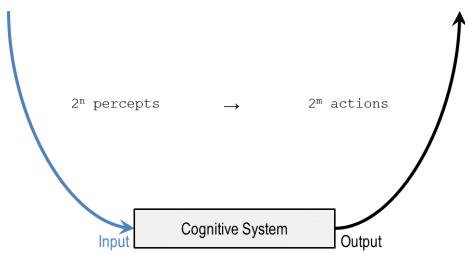


Figure 376: The Challenge of Classification

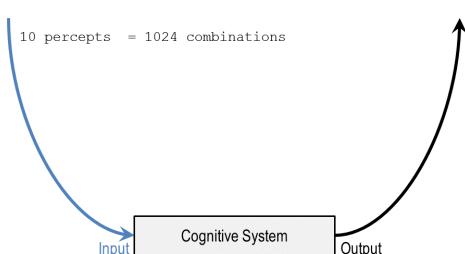


Figure 377: The Challenge of Classification

To see why classification is so powerful and so ubiquitous, and also to understand what exactly is classification, let us go to our overall cognitive architecture for an intelligent agent. This is a diagram that we have come across many, many times. Let us imagine this particular Cognitive System is dealing with a set of percepts. These percepts are in the world. So as an example this Cognitive System may see some object, some animal when the Cognitive System goes to the zoo. This might be an AI agent, or perhaps your friend who goes to the zoo. And look at some animals and there are a large number of percepts in that environment. Has wings, Has feathers, talons, beak and so on. And for the simplicity let's assume that each of the percepts is binary, it's either true or false. So either the animal has wings or doesn't have wings. And depending upon the percepts and the combinations of this percepts one might take different kind of actions. So if it's a friendly animal one might go and pet it, if it's a unfriendly or dangerous animal one might run away from it. So, all kind of actions are possible. Imagine that there are some M actions that are possible. Send, we can again imagine that there is a binary choice here.

So, the total number of combinations of actions, as 2^m to the power m . So as an example. If I have a , if I see a dangerous animal in a zoo, then I might both scream and run away. If I see a friendly animal, I may approach the animal and make cooing noise to the animal. So, a number of actions and combinations of actions are possible. And if n is the number of percepts then I have 2^n to the power n . Combinations of percepts possible. So how is the challenge that the cognitive agent faces? The challenges, that the number of percepts and the number of actions multiplied by the number of combinations of percepts and the number of combinations of actions is very, very large. And we have to map these percepts, combinations of percepts, actions, combinations of actions. This is a very complex mapping. So imagine that only 10 percepts. Image at environment, so I'm looking at an animal and let's take 10 percepts that I'm paying attention to. Then two to the power 10, the number of combination of percepts is 1024, and doing it two to the power of 10 here because I'm assuming each percept has a binary value. If I had 100 percepts, I was not looking at one animal, but I was looking at a scene of animals. Then I may have 100 percepts, in which case I have a much larger number of combinations. And if I had something like 300 percepts, which is not very large number. The number of combinations is, well it's a very large number. more than the combinations of atoms in the universe. Now you and I, and AI agents more generally. Are constantly faced with a complex environment where there are a large number of percepts, and a large number of combinations that are possible. So let's go back to something earlier that we had considered in the class. We defined earlier that, one way of talking what intelligent agent is to think in terms of how can an intelligent agent map percepts into actions. Intelligence is, in part, a lot part perhaps, according to this definition, about action selection. The other aspect of this is. That if the number of perceptions is large and the number of actions is large. Then the mapping between them becomes very large and very complicated quickly. But intelligent agents have only finite resources. How is it then that we can select the

right action. Or at least most of the time select the right action. Even when the environment is very complex. And do so in near real time.

06 - Equivalence Classes

[Click here to watch the video](#)

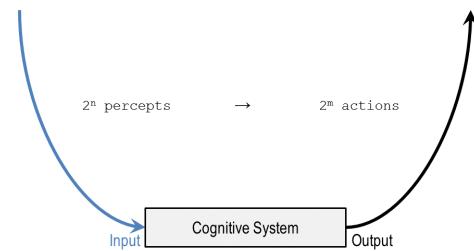


Figure 380: Equivalence Classes

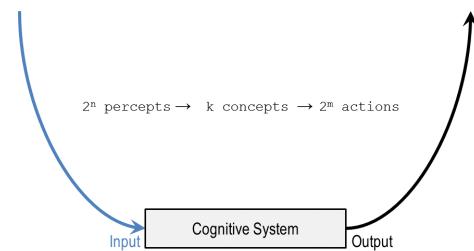


Figure 381: Equivalence Classes

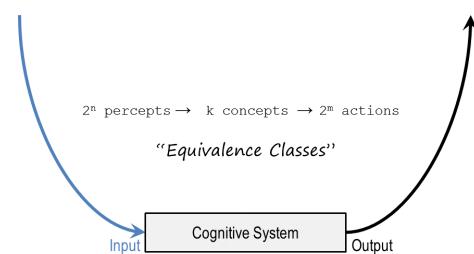


Figure 382: Equivalence Classes

Here is one way in which an intelligent agent can work and accomplish environment with a large number of perception actions possible and yet do so relatively efficiently. Suppose that this 2^n percepts, could be mapped into k concepts where each of these k concept is an equivalence class with a large number of these

combinations. So 2 to the power of n may be very, very large, but k is a much smaller number. So these concepts are now equivalence classes or these percepts. So now instead of indexing my actions on the combinations of percepts, I index my actions on the equivalence classes that are called concepts, and this happens all the time. You go to a doctor, for example and you may go with some signs and symptoms and the doctor says, well, I have to decide whether I'll give you a blue liquid or a red liquid, assume that those are the actions possible. And the actions are now indexed, not as the signs and symptoms or the combination of signs and symptoms, which are potentially very large for human beings, but it's a small number of concepts, the number of diseases, which compared to all the combinations of signs and symptoms, is much smaller. Another example of this that would come from computer programming would be that we might have a small number of different ways in which a computer program can go wrong, by a large number of different percepts that tell us what has actually gone wrong. So, for example in my percepts, I might have whether or not I received a null pointer error, whether or not I received a memory allocation error, whether or not I received an index larger than size of an array error. All of those are different percepts I might have, but it might map the same underlying concept or something that has not yet been initialized. So I'm taking the number of things that I can actually see and mapping them to a smaller number of ways in which the program can actually go wrong. Then instead of having to map each individual percept to some number of actions, I know that if my error is that something has not been initialized I need to find what hasn't been initialized and there's a much smaller list of actions involved looking at each individual variable and seeing where the initialization has not taken place. Now we understand why classification is such an often-studied topic in artificial intelligence. Almost every school of artificial intelligence has studied classification extensively. If there were no concept we were mapping this two to the power n combinations of percepts and use two to the power m combinations of actions,

then we could think of an intelligent agent as one large, giant table. The rows of the table are all the two to the n combinations of percepts that will be that many rows and the columns are through the power m actions. Given a percept, I know exactly what action to take if they would tell you that, and it's going to be a very large table. We don't know how to use it, it will be very costly to use it, and we don't know how to build such a table. What classification is doing is, it is breaking that large table into a large number of small tables, and that's the power of knowledge. When you have knowledge, you can take some complex problem, and break it into a large number of smaller, simpler problems.

07 - Exercise Equivalence Classes

[Click here to watch the video](#)

For each of these three animals, choose the value for each percept that applies to that animal.

Eagle	Bluebird	Penguin
Lays eggs?	Lays eggs?	Lays eggs?
<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has wings?	Has wings?	Has wings?
<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has talons?	Has talons?	Has talons?
<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Flies?	Flies?	Flies?
<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has fur?	Has fur?	Has fur?
<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Large?	Large?	Large?
<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe

Figure 383: Exercise Equivalence Classes

So the next question becomes, given a set of animals, or in general, a set of objects or elements, and a set of percepts for each of those animals, how can we decide what's a good equivalence class for those animals? Consider, for example, the three animals shown here, eagle, bluebird, and penguin. Let us suppose that we knew that there are six percepts that are important for each one of these animals. Lays eggs, has wings, has talons, and so on. But in order to decide what might be a good equivalence class for these three animals, we first have to decide on what might be the right values for each of these percepts for each animal. So I'm going to ask you to use your background knowledge, to fill in the values of the percepts that applies to each of the animals.

08 - Exercise Equivalence Classes

[Click here to watch the video](#)

For each of these three animals, choose the value for each percept that applies to that animal.

Eagle	Bluebird	Penguin
Lays eggs?	Lays eggs?	Lays eggs?
<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has wings?	Has wings?	Has wings?
<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has talons?	Has talons?	Has talons?
<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Flies?	Flies?	Flies?
<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has fur?	Has fur?	Has fur?
<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Large?	Large?	Large?
<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	<input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe

Figure 384: Exercise Equivalence Classes

David, do you know about these three animals? Were you able to answer this question? So, admittedly, I used Wikipedia a little bit to find out some of these answers, but we know that eagles lay eggs, bluebirds lay eggs, penguins. They all lay eggs, and they all have wings. Eagles and bluebirds have talons, whereas penguins kind of more have flippers because they swim, so they don't really have talons. All right. Eagles and bluebirds also fly, whereas penguins don't fly. None of them have fur and we might say that eagles and penguins are large whereas bluebirds are not very large. Our definition of large is kind of variable, it's kind of subjective, but if we were to draw lines somewhere, it's reasonable to say that bluebird is definitely on the smaller end of the spectrum. Good David, you know more about those animals than I do. Imagine that the three animals were given as examples, one after the other. So, we are back to incremental concept learning of the previous lesson. One can use the techniques that we learned in the previous lesson to learn some equivalence class, learn some concept definition with the three animals. But that's not the point here, the point here is not about the learning of the concept, the point here is much more about the nature of the concepts and how they get organized related to each other.

09 - Concept Hierarchies

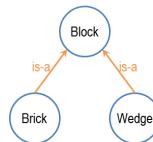
[Click here to watch the video](#)

Figure 385: Concept Hierarchies

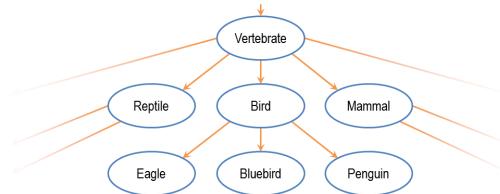


Figure 386: Concept Hierarchies

So, I just talked about the organization of concepts and the relationship of concepts with each other. We only come across this when we were talking about background knowledge in the previous lesson on incremental concept learning. So we had a brick and a wedge. They were both subclasses of a block. In general, the organization of concepts can be much more complex. So here is a set of concepts that you're probably all familiar with. We have different kind of animals, vertebrates, intervertebrates. Vertebrates themselves, can be of different kinds, reptiles, birds, mammals, and so on. And here are birds, which can be of different kinds, eagles, bluebirds, and penguins, like we saw in the previous slide. And generally, organization of these concepts can be a little more complicated. So this is a conceptual hierarchy that I expect all of your family with the animals of various kinds, water breeds and in water breeds, for example. And vertebrates themselves can be a different reptiles, but mammals and birds can be a different kinds, for example, eagles, bluebirds and penguins. So these are classes, and these are sub-classes and super classes. Now the advantage of this kind of conceptual organization is that I can start going in

LESSON 11 - CLASSIFICATION

a top down way establishing and refining each category. So you see an animal and you're trying to decide whether the animal is a bluebird or an eagle or a penguin. And you may start, initially, by saying well, is it a vertebrate? Then, is it a bird? And if it is a bird, then is it a eagle, a bluebird, or penguin? When you and I do processing of this kind, it might happen so fast that we might not be aware of it. But, when we try to put an air-agent, we have to make that processing very, very explicit. Even among humans, imagine for example, what happens when the scientist discovers new kind of species. Then the scientist can begin from the top and say, well, is it a vertebrate? Where exactly in this tree should I put this species? Is it a bird? Is it closer to eagle? Should I put it closer to a bluebird, and so on. So classification of this kind often has a top down, establish, refine, establish, refine connected to it. So one of the benefits of this establish and refine approach is it helps us figure out which variables we need to actually key in on and focus in on. So for example, we saw earlier that eagles are large, but bluebirds are small. So birds can come in different sizes. So if we are trying to establish whether something is a bird, a reptile, or a mamma, we know that mammals also come in different sizes like tiny mice or large elephants. So size doesn't really impact our decision whether it's a reptile, bird, or mammal. But once we've established it's a bird, and we're trying to decide between eagle and bluebird for example, we know that size actually can be something that helps us differentiate between to these two. So now we'll pay attention to size as a variable that matters. So note that there are several things going on here. On one side, there is knowledge of these different classes, but there's also organization of these different classes in a hierarchical particular kind. This organization is very powerful in some ways. If knowledge is power, so is organization. Organization too is power. This organization provides power, because this organization tells you what the controller processing should be. Establish one node, refine it. Establish that node, refine it.

10 - Exercise Concept Hierarchies

[Click here to watch the video](#)

How would you characterize the class 'bird' given the characterization of its subclasses below?

Bird		
Eagle	Bluebird	Penguin
Lays eggs? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Lays eggs? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Lays eggs? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has wings? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has wings? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has wings? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has talons? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has talons? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has talons? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe
Flies? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Flies? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Flies? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe
Has fur? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe	Has fur? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe	Has fur? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe
Large? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Large? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe	Large? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe

Figure 387: Exercise Concept Hierarchies

Let us return to the exercise that we were trying to do earlier, where we had eagle and bluebird and penguin. And we had features, and we had values for the eagle and the bluebird and the penguin. Well, given these three sets of values for the eagle, bluebird and penguin. And given that bird is a superclass of these three classes. What would be the features that you would put in the bird node in that classification hierarchy?

11 - Exercise Concept Hierarchies

[Click here to watch the video](#)

How would you characterize the class 'bird' given the characterization of its subclasses below?

Bird		
Eagle	Bluebird	Penguin
Lays eggs? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Lays eggs? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Lays eggs? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has wings? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has wings? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has wings? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe
Has talons? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has talons? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Has talons? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe
Flies? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Flies? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Flies? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe
Has fur? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe	Has fur? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe	Has fur? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe
Large? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe	Large? <input checked="" type="radio"/> No <input type="radio"/> Yes <input type="radio"/> Maybe	Large? <input checked="" type="radio"/> Yes <input type="radio"/> No <input type="radio"/> Maybe

Figure 388: Exercise Concept Hierarchies

What do you think, David? So what I did, is I looked across the rows and tried to see which features all three birds had in common. They all lay eggs, so I'm going to figure that birds law eggs. They all have wings, so I' going to figure that birds have wings. And none of them have fur, so I'm going to figure that birds don't have fur. Whether or not they have talons can actually vary. So the eagle and bluebird have talons but the penguin does not. And whether

or not they are large can vary. The eagle and penguin are large, but the bluebird's not. So those things don't necessarily define what a bird is. But these three things seem to me like they do. That's a good answer, David. But I should quickly note that this idea that we can decide on the features that should go into a super class, given the features that are shared among these subclasses, works only for certain kind of concept. It doesn't work for all concepts. Particularly, it work for concepts that have a formal nature, as we will see in just a minute.

12 - Types of Concepts

[Click here to watch the video](#)

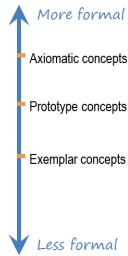


Figure 389: Types of Concepts

We can think of concepts lying on a spectrum. On one extreme end are extremely formal concepts for which we can define logical conditions that are necessary and sufficient for that concept. We'll examine that in more detail in just a minute. On the other end of the spectrum are less formal concepts for which it's hard to define necessary and sufficient conditions. Now here are three points on the spectrum. Axiomatic concepts, prototype concepts, exemplar concepts. There can be other types of concepts as well. We're just going to consider these three concepts because they are the three most common ones. And we'll look at each one of them in turn. In general, humans find it easier to communicate about axiomatic concepts because they are well defined. There is a set of necessary and sufficient conditions that we all agree with. Examples are mathematical concepts. Humans find it harder to communicate about prototype concepts, but most of the time we do quite well.

It's even harder to talk about exemplar concepts like, let's say, beauty or freedom. Similarly, it's easier to teach computers axiomatic concepts or program axiomatic concepts into computers. It's much harder to program or teach prototype concepts. And much, much harder to teach a program exemplar type concepts.

13 - Axiomatic Concepts

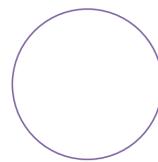
[Click here to watch the video](#)

Axiomatic concepts:
Concepts defined by a formal set of necessary and sufficient conditions.

Example: a circle

Figure 390: Axiomatic Concepts

Axiomatic concepts:
Concepts defined by a formal set of necessary and sufficient conditions.



Circle: all points in a plane that are equidistant from a single point.

Figure 391: Axiomatic Concepts

So let us look at each one of them. Axiomatic concepts, product concepts, and exemplar concept in more detail. Let's begin with axiomatic concept. And axiomatic concept is a concept, it is defined by a formal set of necessary and sufficient conditions. Geometric objects like a circle are good examples. Triangles, squares, rectangles, and so on. So as an example, a circle are all points in a plane that are equidistant from a single point. And the single point is the center of the circle. Now, this is a very formal set of necessary and sufficient conditions. Given any other object, you can see whether or not the particular object is a circle by looking for this particular condition.

14 - Prototype Concepts

[Click here to watch the video](#)

Prototype concepts:
Base concepts defined by a typical example with
overridable properties.

Example: a chair

Figure 392: Prototype Concepts

Prototype concepts:
Base concepts defined by a typical example with
overridable properties.



Figure 393: Prototype Concepts

Prototype concepts:
Base concepts defined by a typical example with
overridable properties.



```
Chair
#-of-legs : four
material : metal
has-back : true
has-arms : false
is-cushioned : false
...
```

Figure 394: Prototype Concepts

Prototype concepts:
Base concepts defined by a typical example with
overridable properties.

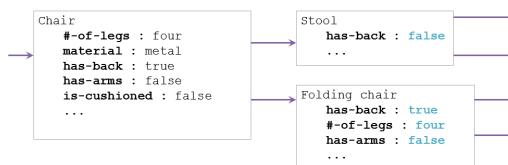


Figure 395: Prototype Concepts

The notion of axiomatic concepts is the classical view in cognitive systems. Here's an alternative view. This is called Prototypical concepts. Prototypical concept is a base concepts defined by some typical properties that can sometimes be overridden. So, an example is a chair. You and I have a notion of a prototypical chair. Our notion of a prototypical chair may have, for example, there is a back, there is four legs and so on. So, here might be your and my notion of a prototypical chair. It has a back, it has a seat, it has four legs and so on. Now I can represent this notion of the prototypical chair in the language of frames. This is something we have come across earlier in the class. A frame has slots and fillers, as you may recall, and we use frames to represent stereotypes. Here we're talking about prototypes, so very closely related. So concept is the content you can represent. Frame is the form in which we can represent it. So a notion for the prototypical chair might be, it has four legs, it, the material is metal, it has a back, it does not have arms, and it's not cushioned. Note that these are the typical properties of a chair. Of course, some chairs need not necessarily satisfy all of these properties. That is why there are no, no necessary and sufficient conditions here. For example, we may come across a chair. Which is made of wood. While you would still consider it a chair, even if you're not strictly satisfied with this particular definition. Thus, these properties can be overridden in case of specific instances. But we'll still have the basic notion of a prototypical chair so that we can in fact communicate with each other about what a chair is. Despite the fact that we can override these properties we do still have a notion of prototypical chair. So the relationships between concepts and frames actually is quite close. Recall that when we were talking about frames, we were also talking about inheritance and default. The notion of default in frames is closely connected to the notion of typical properties and concepts. So chair has this prototypical notion with some of these typical properties, and we can think of this as having a default values. By default, we assume the number of legs is four, the material is metal and so on. Here is a stool and this stool is a kind of chair

which means that it inherits all the values and all the slots that are there in the chair directly except for those that happen to be different from the one in the chair. So then an example here it over writes a notion that the chair necessarily has to have a back. In the case of a stool, the stool does not have a back.

15 - Exemplar Concepts

[Click here to watch the video](#)

Exemplar concepts:
Concepts defined by implicit abstractions of instances, or exemplars, of the concept.
Example: beauty



Figure 397: Exemplar Concepts

Like so many concepts with refinement necessary in sufficient conditions, for typical concepts in typical conditions, what about exemplar concepts? Exemplar concepts don't even have typical conditions, let alone necessary and sufficient conditions. In case of exemplar concepts, I can give you examples. Perhaps I can do some implicit abstraction of some examples, but it's about as far as I can go. Consider the example of beauty for a second. There are four examples of something beautiful. Here's a painting by Van Gogh, here's a beautiful sunset, a beautiful flower, a beautiful dance and so on. While I can give examples of the concept of beauty, it's really hard to come up with the typical condition

of a beauty. Exemplar concepts are very hard to define. And for that reason, they are also very hard to communicate to each other, or to teach in a art program. Exemplar concepts can be culture specific, sometimes even individual specific.

16 - Order of Concepts

[Click here to watch the video](#)

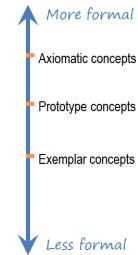


Figure 398: Order of Concepts

Figure 396: Exemplar Concepts

To summarize then, concepts can be of many different kinds. From very formal concepts, like Axiomatic concepts, to less formal concepts, like Exemplar concepts. Of course, we can, if we want less formal concepts, like exemplar concepts, philosophers often talk about concepts called qualia, Q-U-A-L-I-A. Qualia for the raw sensations that we may get from our sensors. An example of qualia is, bitterness. I'm sure you've come across some bitter fruit, some time or the other. And you can even taste it, inside your mouth right now, if you wanted to. But it's very hard to communicate what a qualia is to anyone else, what. Your notion of a genesis to anyone else.

17 - Exercise Order of Concepts

[Click here to watch the video](#)



Figure 399: Exercise Order of Concepts

Let us do an exercise together. On the left again is a spectrum from very formal to less formal. In the right here are six concepts. Inspirational, reptile, foo. Foo here is the same concept that we came across when we're talking about incremental concept learning. This is the same foo here. Right triangle, holiday, saltiness. Can you rank the six concepts according to the notion of formality that we have studied so far?

18 - Exercise Order of Concepts

[Click here to watch the video](#)



Figure 400: Exercise Order of Concepts

So part of the point of looking at these two different kinds of concepts is that depending on the different kinds of concepts we are dealing with right now, you may come up with a different knowledge representation and a different inference method. Let me explain. Supposing we're dealing with concepts like Foo or Holiday or Inspirational. Then the case-based reasoning method might be a very good method for dealing with things of that kind. We may have experience with specific holidays, but we cannot abstract them out into a concept with prototypical conditions. On the other hand, if we were dealing with concepts like a right triangle or a reptile, for which we can define necessary and sufficient conditions, in that case there are alternative methods available that might be more suitable than case-based reasoning. So instead of thinking in terms of one method that is going to work for all conditions and all concepts, we might want to think in terms of a array of methods, where each method is more or less useful for different kind of conditions or different kind of concepts. David, I am sure you recall

mentally who was a Russian chemist who came up with the basic notion of the chemical periodic table. I'm sure all the students in the class know about the chemical periodic table, which organizes all the elements according to certain properties like hydrogen, oxygen, calcium, and so on. Now, Mendeleev came up with this notion of a chemical periodic table, and in some sense what we're trying to do in this course is to build a similar kind of periodic table, except that this is a periodic table of the elements of mind. It's a periodic table of the basic fundamental elements that compose intelligence. Instead of talking about elements and valances and atoms and so on, what we are going to be talking about are methods and representations. So it is as if we are discovering the fundamental knowledge for presentations and organizations and reasoning methods that go with it. Case based reasoning was one. Reasoning method that went with certain kinds of concepts that are hard to abstract and into conditions, like typical conditions, are necessarily logical conditions.

19 - Bottom-Up Search

[Click here to watch the video](#)

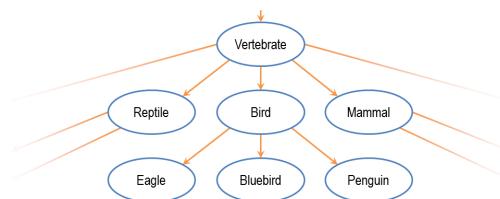


Figure 401: Bottom-Up Search

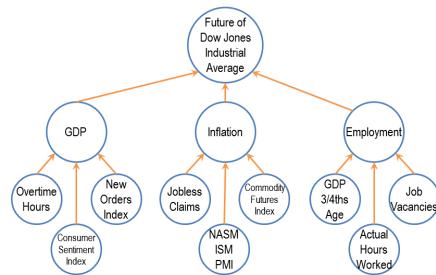


Figure 402: Bottom-Up Search

Let us build on this metaphor of periodic table a little bit further. So earlier we came across another method of dealing with classification, which we call top down or establish define. In that method we had this classification hierarchy which start with a concept, will establish it, then refine it, and refine it further if needed. That particular control of processing is very well suited for one kind of organization of concepts. It's very well suited for one set of situations where we know something is already a vertebrate, we're trying to establish whether it's a bird or a bluebird. And a different kind of classification task. A better control of processing is to go bottom up. Let's look at this a little bit more carefully. Here are a number of leaf nodes. And we know, the agent knows something about the value for each of these leaf nodes. And the task is to make a prediction at the root node. So in this particular case imagine the task of the AA agent is to predict the of the Dow Jones Industrial Average tomorrow. It'll be great to have an AA agent like that. If we had a good AA agent like that, you and I could both become very rich. Now how could this A, agent make a prediction about the Dow Jones Industrial Average tomorrow? Well, one way in which it could do it is it could look at the. Information it has about the GDP the inflation and employment today. but how does it know about the value GDP or the inflation or employment today. Well it can look then at the values of the overtime hours. The consumer sentiment index, new orders index and so on and so forth. Now, the processing is largely, bottom up. We know something about the values or the features that go into this concept. And they're going to this concept, they're going to this concept. You can abstract them and find the value of the GDP. And similarly for this, and then abstract it further. So the control of processing in this particular case, we might call it identify and abstract, identify an abstract. Bottom-up controller processing rather than top-down in the previous case. We have just defined two other elements of our periodic table, of our growing periodic table of intelligence. In this latter element, the bottom-up classification, the conditions of application are different.

20 - Assignment Classification

[Click here to watch the video](#)

Assignment

How would you use classification to design an agent that could answer Raven's progressive matrices?

Figure 403: Assignment Classification

How would you apply the principles of classification, to designing an agent that can solve Raven's progressive matrices? In answering this there's a lot of questions to touch on. For example, will you develop the classification scheme yourself, or are you going to have your agent learn it as it encounters new problems. So what would that classification scheme look like? What percepts will the agent use to classify new problems into that classification scheme? Then once it's classified them, how will that classification actually help it solve the problem? What will it be able to do this way that it wouldn't have been able to do otherwise?

21 - Wrap Up

[Click here to watch the video](#)

To recap...

- Concept learning
- Equivalence classes
- Concept hierarchies
- Axiomatic, prototypical, and exemplar concepts
- Bottom-up search

Figure 404: Wrap Up

So today we've talked about classification, which is one of the biggest problems in AI. We started by revisiting incremental concept learning and reminding ourselves how it allowed us to take examples and abstract away a concept. We then looked at the idea of equivalence classes and

how we can group sets of percepts into equivalence classes to establish a particular instance of a concept. Within this is the hierarchies of concepts, such as the animal kingdom, where animals can grow kind of into vertebrae, birds and penguins. We then discuss the idea of different types of concepts like axiomatic or exemplar concepts, and how each of them have different definitions and different affordances. Finally, we discuss bottom-up search, so instead of establish and refine, we look at the lower level variables and abstract up from them. Next, we're going to move on to logic, which is a little bit unrelated to this. But if you're interested in classification, you can look ahead to all our different lessons on design, such as diagnosis and configuration. They're going to really heavily leverage our idea of classification.

22 - The Cognitive Connection

[Click here to watch the video](#)

One could say a lot about the connection between classification and cognition. This is because classification is ubiquitous in cognition. You're driving a car on the street, you see a friend driving his car, you look, take a look at the car, and you see a Porsche. Classification. You're on a computer program, the output is faulty. You look at the output, decide on the bug. You name the bug. Classification. You go to a doctor with certain signs and symptoms, the doctor names a disease category, classification. The reason classification is so ubiquitous is because it allows us to select actions. Once the doctor knows what the disease category is, he can suggest a therapy. Once you know what

the bug is you can decide on a repair for that bug. If action selection indeed is a very productive characterization of intelligence then we can see why classification is central to cognition.

23 - Final Quiz

[Click here to watch the video](#)

All right. Please right down what you learned in this lesson in this box for us to peruse later.

24 - Final Quiz

[Click here to watch the video](#)

And thank you for doing it.

Summary

Classification is ubiquitous in human cognition where humans continuously and constantly perform classification in the day-to-day life.

References

1. Stefk, M. Introduction to Knowledge Systems, Pages 543-556, 558-596.

Optional Reading:

1. Stefk, Chapter 7, Part 1 T-Square Resources (Stefk- Classification Part1_Pgs 543-556_.pdf)
2. Stefk, Chapter 7, Part 2 T-Square Resources (Stefk Classification Part 2_Pgs 588-596_.pdf)

Exercises

None.

Lesson 12 - Logic



If, dear Reader, you will faithfully observe these Rules, and so give my little book a really fair trial, I promise you, most confidently, that you will find Symbolic Logic to be one of the most, if not the most, fascinating of mental recreations! – Lewis Carroll, Introduction to Symbolic Logic.

01 - Preview

[Click here to watch the video](#)

Planning

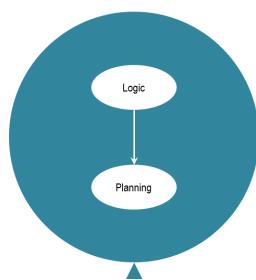


Figure 405: Preview

talking about a formal notation for writing sentences in logic. This formal notation will have things like conjunctions and disjunctions. Then we'll talk about truth tables, a topic that you probably already know. We'll talk about rules of inferences like modus ponens and modus tollens. Finally we'll discuss methods for proving theories [proving theorems by refutation] by repetition. One of those methods is called, resolution theorem proving.

02 - Why do we need formal logic

[Click here to watch the video](#)

Why do we need formal logic?

Lesson Preview

- Formal notation
- Conjunctions, disjunctions, negations, implications
- Truth tables
- Rules of inference
- Resolution theorem proving

Soundness: Only valid conclusions can be proven.

Completeness: All valid conclusions can be proven.

Figure 407: Why do we need formal logic

Figure 406: Preview

Today we'll discuss logic. Logic is a formal language that allows us to make assertions about the world in a very precise way. We learn about logic both because it is an important topic and also because it forms the basis of additional topics such as planning. We'll start

Let us begin by asking ourselves, why do we need use formal logic to design an AI agent? The way the formal logic in an AI agent would work is that there will be two parts to the AI agent. The first part of the agent will consist of a knowledge base. The knowledge will contain the agents knowledge all over the world. That knowledge

will be represented in sentences in the language of logic. The second part will consist of an inference engine. The inference engine will apply rules of inference to the knowledge that the agent has. So remember again, two parts, the knowledge base and then rules of inference. Now there are certain situations in which we want the AI agent to be able to show, to be able to prove that the answers, that it derives to any problem in fact are provably correct. If we want to show this to itself, or to other users. In other situations, we may want the AI agent to generate only provably correct solutions. How can we guarantee that? Well we need two things. First, we need a complete and correct knowledge base. And second, we need rules of inference that will give guarantees of correctness of the answer. The inference then has two parts to it. The first part is called soundness. The property of soundness means that rules of inference will derive only those conclusions that are, in fact, valid. The second property is completeness. The property of completeness means that the [AI agent] will derive all of the valid conclusions. These are two very important properties to have. If an AI agent can use logical rules, logical rules of inference on its knowledge base and provide guaranteed soundness and completeness, then it's a very useful thing for an AI agent to have. For this reason, logic has been an important part of AI since the inception of the field. In fact it continues to be an important part of research and modeling AI. In this course however we'll discuss logic only to a very limited degree. There are two reasons for this. First, our priorities in this course are a little different. Instead of talking about knowledge in the form of logical sentences, we are much more interested in conceptual knowledge and experiential knowledge and heuristic knowledge. Second, recall that we said that a logic based agent has two parts to it, the knowledge base and the rules of inferences. Even if the rules of inferences are, in fact, guaranteed to be sound and complete, there is a problem about how do we construct a correct and complete knowledge base? If the knowledge base of an AI agent is not correct and complete, then it may not give you useful answers, even if the

roots of inference are sound and complete. Thus, in this course, we'll use logic only to the degree to which it is useful for specifying other methods. Methods that use conceptual knowledge or experiential knowledge or heuristic knowledge.

03 - Inferences About Birds

[Click here to watch the video](#)

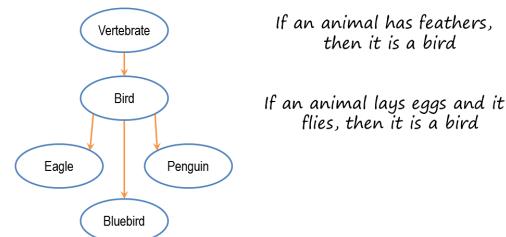


Figure 408: Inferences About Birds

Now we have come across this particular kind of problem earlier. This was a classification hierarchy. Vertebrates can be of different kinds. Bird is one kind. Birds can be a different kind. Eagle, bluebird, penguin are three classes of birds. Now imagine we have knowledge like, if an animal has feathers then it is a bird. When we're discussing this classification hierarchy we had tried to define the concept of bird, and we had said at that time that if an animal has feathers it is a bird, and if an animal lays eggs and it flies then it's a bird. It is sentences like this that we'll try to put in the language of logic.

04 - Exercise Inferences About Foos

[Click here to watch the video](#)

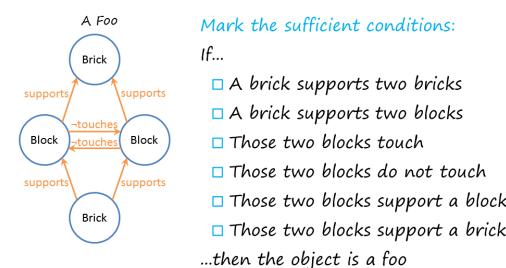


Figure 409: Exercise Inferences About Foos

Before we represent those sentences in the language of logic, let us consider another example of conceptual knowledge and its relationship to logic first. So here is a concept of four. We have come across this earlier. They were a block and a block and a brick at the bottom and a brick at the top. And some relationship between these objects. Given this conceptual knowledge about foo we can ask ourselves, what are the sufficient conditions for something to be a foo? Here are several choices. Please mark all of those choices that together make for sufficient conditions for the concept of foo

05 - Exercise Inferences About Foos

[Click here to watch the video](#)

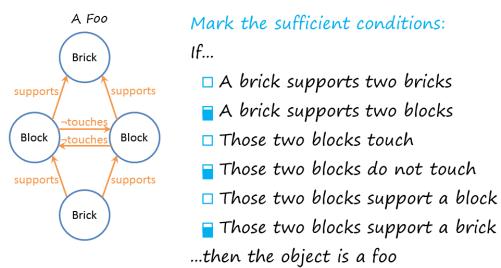


Figure 410: Exercise Inferences About Foos

What do you think, David? So, from the bottom, we have the brick has to support two blocks. Not necessarily two bricks. We know that those can be any kind of block. We also have that those two blocks cannot touch. So, that condition is important. And we also have that those two blocks must then support a brick. And when we learned earlier, it's not sufficient for that to be any kind of block. The block on top has to be a brick as well. So, if these three conditions. Are as efficient to find a Foo in terms of logic. That is good David. So what we are learning here is given conceptual knowledge how can we translate it into the language of logic?

06 - Predicates

[Click here to watch the video](#)

Predicate:
A function that maps object arguments to true or false values

If an animal has feathers,
then it is a bird

Feathers(animal)

If Feathers(animal):
Then Bird(animal)

Figure 411: Predicates

Recall that we said that a lot of this AI agent will have two parts to it, a knowledge base and then the rules of inference. We'll come to the rules of inference a little bit later. First let us look at how can we construct a knowledge base in the language of logic. So what we are trying to do now is that an AI agent has some knowledge about the world and it is going to express it in the scheme of logic. In earlier schemes of knowledge representation, we discussed how there were objects and relationships between objects. And any knowledge representation scheme we need to capture both objects and relationships between those objects. Logic has a particular way of doing it. So consider an object like a bird. This object may have various arguments. We'll define something called a predicate, which is a function that maps object arguments to either true or false. So let us consider an example. Here we have bluebird as the object and feathers as the predicate on this object. Let's consider this example. Here, bluebird is the object and feathers is the predicate on this object. Feathers is now a function that can map either into true or into false. Either bluebird has feathers or bluebird doesn't have feathers. In this particular case, feathers of bluebirds would be true, because bluebirds do have feathers. Now, just like we had bluebird as the object in the previous example, here we have animal as the object, the same predicate. Now, of course, not all animals will have feathers, so this particular predicate may be true or false, depending on the choice of the animal. In this sentence there are two predicates, one object, animal still, but there is a predicate feathers and a predicate bird. And we can capture the relationship between these two

predicates, by saying that if feathers animal is true, then bird animal is also true. This example has two predicates. Here there's one object, the animal. But the predicates are feathers and birds. And the sentence is capturing a relationship between the two predicates. If the animal has feathers, then the animal is a bird. In logic we call sentences like this as having an implication. This is an implicative relationship. So in logic, we'll read this as Feathers(animal) implies Bird(animal). Or if the animal has feathers, then it implies that the animal is a bird.

07 - Conjunctions and Disjunctions

[Click here to watch the video](#)

If an animal lays eggs and it flies, then it is a bird

$$\text{Lays-eggs(animal)} \wedge \text{Flies(animal)}$$

If Lays-eggs(animal) \wedge Flies(animal):
Then Bird(animal)

Figure 412: Conjunctions and Disjunctions

If an animal lays eggs or it flies, then it is a bird

$$\text{Lays-eggs(animal)} \vee \text{Flies(animal)}$$

If Lays-eggs(animal) \vee Flies(animal):
Then Bird(animal)

Figure 413: Conjunctions and Disjunctions

If an animal flies and is not a bird, it is a bat.

$$\text{Flies(animal)} \wedge \neg\text{Bird(animal)}$$

If Flies(animal) \wedge $\neg\text{Bird(animal)}$:
Then Bat(Animal)

Figure 414: Conjunctions and Disjunctions

Now, consider another sentence that we have come across earlier. If an animal lays eggs, and it flies, then it is a bird. How do we write this in the language of logic, given that there is conjunction here. So this time, we can have two predicates again. There is a predicate of lays-eggs, coming from here. The predicate of flies, coming from here. And we can denote a conjunction between them. Which in the language of logic is often put in this form. Now we can re-write this sentence in the following form. If the animals lays eggs and the animal flies, then the animal is a bird. Remember again, this semi colon here, really is denoting implication for now. Remember again, that in logic, this really stands for an implication. Consider the slightly different sentence. Suppose if the sentence was if an animal lays eggs or it flies it is a bird. In that case, again, we'll have two predicates, but this time we'll have a disjunction between them. And the sentence would become or if animal lays eggs or animal flies, then the animal is a bird. And again, this is an implication. Let us continue with the our exercise in which we are learning how to write sentencable language of logic. It is under the sentence, if an animal flies and is not a bird. So, it is a negation here, then it is a bat. How do we write that in logic? So I'm still interested in writing the antecedent of this particular sentence, and I may be able to say that animal flies is a conjunction here, because it is an and here, and we have this negation symbol for this predicate, bird. Now we can write a complete sentence by saying that the animal flies, conjunction. Animal is not a bird, implies animal is a bat.

08 - Implies

[Click here to watch the video](#)

If Lays-eggs(animal) \wedge Flies(animal):
Then Bird(animal)

Lays-eggs(animal) \wedge Flies(animal) \Rightarrow Bird(animal)

Figure 415: Implies

Now, I have talking a little about implication. Let's see how do we actually write, implication and logic. So here is a sentence, if animal lays eggs and animal flies, it is implication is that the animal is a bird. In logic we write this using the symbol, arrow symbol, or an indication, so if the animal lays eggs and animal flies, implication animal is a bird. So here is the left hand side of the implication, here is the left hand side of the implication. The left hand side of the implication, implies the right hand side

09 - Notation Equivalency[Click here to watch the video](#)

Operator	Symbol	Accepted Symbol
AND	$A \wedge B$	$A \& B$ $A \&& B$
OR	$A \vee B$	$A B$ $A B$
NOT	$\neg A$	$!A$ $\sim A$
IMPLIES	$A \Rightarrow B$	$A = B$ $A == B$ $A > B$

Figure 416: Notation Equivalency

Generally speaking, you won't have these symbols on your keyboard. You can find them in your character map and you are welcome to use them if you'd like to. But for the exercises in the rest of this lesson and in the next lesson, feel free to use the symbols given over here. These are the symbols for AND, NOT, OR and Equals that come from Java or Python. So, feel free these when you are doing the exercises that you'll come across in the rest of this lesson.

10 - Exercise Practicing Formal Logic[Click here to watch the video](#)

If an animal lays eggs and does not have feathers, it is a reptile.
Lays-eggs(animal) \wedge \neg Feathers(animal) \Rightarrow Reptile(animal)

If an animal has feathers or has talons, it is a bird.
Feathers(animal) \vee Talons(animal) \Rightarrow Bird(animal)

If an animal lays eggs, has a beak, and flies, it is a duck.
Lays-eggs(animal) \wedge Beak(animal) \wedge Flies(animal) \Rightarrow Duck(animal)

If an animal lays eggs, has a beak, and do not fly, it is a platypus.
Lays-eggs(animal) \wedge Beak(animal) \wedge \neg Flies(animal) \Rightarrow Platypus(animal)

Figure 417: Exercise Practicing Formal Logic

So remember we are still trying to learn how to build a knowledge based on the language of logic. To put it all together, consider four exercises. Here is the sentence. Please put it in the language of logic. Similarly for this sentence, this sentence, this sentence.

11 - Exercise Practicing Formal Logic[Click here to watch the video](#)

If an animal lays eggs and does not have feathers, it is a reptile.
Lays-eggs(animal) \wedge \neg Feathers(animal) \Rightarrow Reptile(animal)

If an animal has feathers or has talons, it is a bird.
Feathers(animal) \vee Talons(animal) \Rightarrow Bird(animal)

If an animal lays eggs, has a beak, and flies, it is a duck.
Lays-eggs(animal) \wedge Beak(animal) \wedge Flies(animal) \Rightarrow Duck(animal)

If an animal lays eggs, has a beak, and do not fly, it is a platypus.
Lays-eggs(animal) \wedge Beak(animal) \wedge \neg Flies(animal) \Rightarrow Platypus(animal)

Figure 418: Exercise Practicing Formal Logic

Okay David, what did you have for the first sentence? So for the first sentence, I created the predicates lays eggs, feathers, and reptile, and said if the animal lays eggs and the animal does not have feathers that implies the animal is a reptile. For the second one, I said if the animal has feathers or the animal has talons that implies that the animal is a bird. So animal feathers or talons, feathers or talons. For the third, which is a longer one, I said, if the the animal lays eggs and the animal has a beak and the animal flies, all three of these all in a chain, that implies the animal is a duck. And similar for the fourth one, we have three predicates on the left side. If the animal lays eggs, the animal has a beak, and the

animal does not fly, then the animal is a Platypus. This guy shows us why a platypus is such a strange animal because we need to make a lot of caveats in order to find what a platypus really is. Good David, that looks right to me. So to wrap this part up, let us note that when we defined what a predicate was, we set up a predicate like Flies can map into true or false. Well okay, a predicate can map into true or false. What about complicated sentences like this, which are multiple predicates, as well as, implications? How do we find out whether the sentence as a whole maps into true or false? That's what we're going to look at next. We're looking at truth tables.

12 - Truth Tables

[Click here to watch the video](#)

A	B	$A \vee B$
True	True	True
True	False	True
False	True	True
False	False	False

Figure 419: Truth Tables

So we'll now build truth tables for conjunctions and disjunctions and negations of sentences, so that we can find the truth of complex sentences stated in logic. Now many of you probably are familiar with truth tables, and if you are in fact familiar with truth tables, then you can skip this part and go directly to implication elimination. If you're not familiar with this then please stay with me, but even so I'm going to go through this quite rapidly. So here is the truth table for A or B. If A is true, then B. If A is true and B is true, then A or B is true. If A is true and B is false, then A or B is still true, because A was true. If A is false and B is true, then A or B is true, because B was true. One of them is true, makes this true. If A is false and B is false, than A or B is false.

13 - Exercise Truth Tables I

[Click here to watch the video](#)

A	B	$A \vee \neg B$
True	True	True
True	False	True
False	True	True
False	False	False

A	B	$\neg A \wedge \neg B$
True	True	False
True	False	False
False	True	False
False	False	True

Figure 420: Exercise Truth Tables I

Let us try a couple of simple exercises. So here we have A, B and we want to find a truth value of A or not B. Given these values for A and B, can you please write down the truth values for A or not B. And similarly, for not A and not B

14 - Exercise Truth Tables I

[Click here to watch the video](#)

A	B	$A \vee B$
True	True	True
True	False	True
False	True	True
False	False	False

A	B	$A \vee \neg B$
True	True	True
True	False	True
False	True	False
False	False	True

A	B	$\neg A \wedge \neg B$
True	True	False
True	False	False
False	True	False
False	False	True

Figure 421: Exercise Truth Tables I

So for A or not B, I got that if A is ever true, then this has to be true, because it's A or not B. When A is false the negation flips the value of B, so it makes it true when B is false, but keeps it false when B is true. For not A and not B, that means that any time either A or B is true, then this is all false. So when A is true, this is false. When B is true, this is false. When both are false, this becomes true, because those negations flip the values of both A and B.

15 - Exercise Truth Tables II

[Click here to watch the video](#)

A	B	C	$A \vee (B \wedge \neg C)$
True	True	True	<input type="text"/>
True	True	False	<input type="text"/>
True	False	True	<input type="text"/>
True	False	False	<input type="text"/>
False	True	True	<input type="text"/>
False	True	False	<input type="text"/>
False	False	True	<input type="text"/>
False	False	False	<input type="text"/>

Figure 422: Exercise Truth Tables II

Now, we can play the same game, for ever more complex sentences. So, here I've again, three predicates, A, B and C. And here's a more complicated sentence that involves all three of those predicates. A or B, and within parentheses, B and, nought C. And we can find the truth values for this particular, sentence, given the truth values for the predicates A, B and C. Why don't you give it a try and write down the values here?

16 - Exercise Truth Tables II[Click here to watch the video](#)

A	B	C	$A \vee (B \wedge \neg C)$
True	True	True	<input type="text"/> True
True	True	False	<input type="text"/> True
True	False	True	<input type="text"/> True
True	False	False	<input type="text"/> True
False	True	True	<input type="text"/> False
False	True	False	<input type="text"/> True
False	False	True	<input type="text"/> False
False	False	False	<input type="text"/> False

Figure 423: Exercise Truth Tables II

David, what did you come up with? From the beginning, we start with A or something. So as long as A is true, then we know the result is true. We don't even care about the rest of it. When A is false, then we need to look at B and the not C. Because this is an and, we need both B and not C to both be true. So, when B is false, we can go ahead and say this is false over here. And when not C is false, we can go ahead and say this is false over here. Not C is false, if C is true, so this is also false. So, our only other answer that's true is that row right there. So, as

you can see, this can become very complicated very quickly. But David did get the answer to the truth value of this particular sentence, based on the truth values of the predicates that are inside the sentences. So in principle now, we can see how we can compute the truth value of very, very complicated sentences returning logic.

17 - Exercise Commutative Property[Click here to watch the video](#)

Commutative Property			
A	B	$A \wedge B$	$B \wedge A$
True	True	<input type="text"/>	<input type="text"/>
True	False	<input type="text"/>	<input type="text"/>
False	True	<input type="text"/>	<input type="text"/>
False	False	<input type="text"/>	<input type="text"/>

Figure 424: Exercise Commutative Property

The construction of these truth tables, allows us to illustrate certain important properties of logical predicates. To see those properties, let us do an exercise together. So here we have the predicate A, and the predicate B. And here we have A and B, and B and A. Please fill these boxes, the truth values of A and B, the truth values of B and A.

18 - Exercise Commutative Property[Click here to watch the video](#)

Commutative Property			
A	B	$A \wedge B$	$B \wedge A$
True	True	<input type="text"/> True	<input type="text"/> True
True	False	<input type="text"/> False	<input type="text"/> False
False	True	<input type="text"/> False	<input type="text"/> False
False	False	<input type="text"/> False	<input type="text"/> False

Figure 425: Exercise Commutative Property

That's good, David. And as you know, this property is called the commutative property. The commutative property says that the truth

value for A and B is the same as the truth value for B and A. So whenever I have A and B, and can re-write it as B and A.

19 - Exercise Distributive Property

[Click here to watch the video](#)

			Distributive Property	
A	B	C	$A \wedge (B \vee C)$	$(A \wedge B) \vee (A \wedge C)$
True	True	True	<input type="text"/>	<input type="text"/>
True	True	False	<input type="text"/>	<input type="text"/>
True	False	True	<input type="text"/>	<input type="text"/>
True	False	False	<input type="text"/>	<input type="text"/>
False	True	True	<input type="text"/>	<input type="text"/>
False	True	False	<input type="text"/>	<input type="text"/>
False	False	True	<input type="text"/>	<input type="text"/>
False	False	False	<input type="text"/>	<input type="text"/>

Figure 426: Exercise Distributive Property

Let us try a slightly more complicated exercise. This time, we have three variables, A, B, and C. And here are the combinations of the truth values of A, B, and C. Here on the right are two formulas. The first one says, A and parenthesis B or C parenthesis closed. The second says parenthesis A and B parenthesis closed or parenthesis A and C parenthesis closed. Please write down the truth values for these two formulas.

20 - Exercise Distributive Property

[Click here to watch the video](#)

			Distributive Property	
A	B	C	$A \wedge (B \vee C)$	$(A \wedge B) \vee (A \wedge C)$
True	True	True	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
True	True	False	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
True	False	True	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
True	False	False	<input type="checkbox"/>	<input type="checkbox"/>
False	True	True	<input type="checkbox"/>	<input type="checkbox"/>
False	True	False	<input type="checkbox"/>	<input type="checkbox"/>
False	False	True	<input type="checkbox"/>	<input type="checkbox"/>
False	False	False	<input type="checkbox"/>	<input type="checkbox"/>

Figure 427: Exercise Distributive Property

Did you write down the truth values for these two formulas, David? I did. And on the one on the left, we find it's little bit easier because it starts with A and, which means anytime A is false, we can go ahead and write that this is false. When A is true, that means we can ignore

it and evaluate solely based on B or C. Whenever either of them is true, the result is true. And only when both of them are false, is the result false. So the one on the right is a little bit more complicated because for each of them we need to look at both A and B and A and C. But because there's an or in the middle, that means that once we discover that one of them is true, we don't need to really worry about the other one. A and B is true whenever both A and B are true, and so A and B are true here and here, meaning we can go ahead and say that those are true. We don't need to look at A and C here. Similarly, A and C are true here. So we can go ahead and say that this one is true as well. For all the ones at the bottom, A is false which means it's going to go ahead and render both of them false. And for the fourth one, B and C are both false, meaning each individually is going to become false. So what we see here is that the truth values for these two formulas actually end up the same, so I guess they're equivalent as well. Good, David. So this property's called the distributive property. The distributive property says that these two formulas have the same truth values. And in particular, if there is B and C inside the parentheses with a disjunction between them, and A is outside the parentheses with a conjunction between them, then we can move A inside the parentheses by first writing A and B and then writing A and C and taking the disjunction of the two. We can also think of this as distributing the part outside of both the predicate and the operator into both the ones on the inside. We take the A and apply it to B, so A and B. We take the A and apply it to C, so A and C. And we preserve the operator in between B and C in between the two new parenthesis. So if this had been A or B or C, this would become A or B, or B or C. This would become A and B and C and would be all the operators here.

21 - Exercise Associative Property

[Click here to watch the video](#)

Associative Property				
A	B	C	$A \vee (B \vee C)$	$(A \vee B) \vee C$
True	True	True	<input type="text"/>	<input type="text"/>
True	True	False	<input type="text"/>	<input type="text"/>
True	False	True	<input type="text"/>	<input type="text"/>
True	False	False	<input type="text"/>	<input type="text"/>
False	True	True	<input type="text"/>	<input type="text"/>
False	True	False	<input type="text"/>	<input type="text"/>
False	False	True	<input type="text"/>	<input type="text"/>
False	False	False	<input type="text"/>	<input type="text"/>

Figure 428: Exercise Associative Property

Let us do one of their exercising in two tables illustrate one of the property of logical predicate. Again here are three predicates, and here are two formulas. It should be a simple exercise. Please write down the truth values, of the two formulas, in these boxes.

22 - Exercise Associative Property

[Click here to watch the video](#)

is called the associative property. The associative property simply says that we can change the location of the parentheses here, A or parentheses B or C, parenthesis close, has the same value as A or B parenthesis close or C. So that we can change the location of the parentheses. The same would have been true if these were both conjunctions. So if it was A conjunction parentheses B conjunction C or A conjunction B, conjunction C, it would have had the same values. The difference between these formulas and the ones we were doing before are the values of these operators. Associative property works when it's both ors or both ands. Distributive property worked when there was a mixture of operators.

23 - Exercise de Morgans Law

[Click here to watch the video](#)

Associative Property				
A	B	C	$A \vee (B \vee C)$	$(A \vee B) \vee C$
True	True	True	<input type="text"/> True	<input type="text"/> True
True	True	False	<input type="text"/> True	<input type="text"/> True
True	False	True	<input type="text"/> True	<input type="text"/> True
True	False	False	<input type="text"/> True	<input type="text"/> True
False	True	True	<input type="text"/> True	<input type="text"/> True
False	True	False	<input type="text"/> True	<input type="text"/> True
False	False	True	<input type="text"/> True	<input type="text"/> True
False	False	False	<input type="text"/> False	<input type="text"/> False

Figure 429: Exercise Associative Property

What did you get, David? So like before, because this is an or, as soon as we see that A is true, we can go ahead and write down true for all of these. When A is false, we just need to evaluate B or C. When B is true, we know it's already true. And then if B is false we need to look at C. If C is true, then it's still true, and if C is false, then all three are false and that's the only time it's false. Over here, we've moved the parentheses but really, we're not changing what we do. We can think of it as stating at the bottom with C being true. Anytime C is true, the entire thing is true. So C is true here, here, here, and here. If that evaluates to false, we need to look at A or B, and so on. So we end up doing the exact same process. That's good, David. And this property

de Morgan's Law

A	B	$\neg(A \wedge B)$	$\neg A \vee \neg B$
True	True	<input type="text"/>	<input type="text"/>
True	False	<input type="text"/>	<input type="text"/>
False	True	<input type="text"/>	<input type="text"/>
False	False	<input type="text"/>	<input type="text"/>

Figure 430: Exercise de Morgans Law

One other property of logical predicates that we will see very soon in action is called de Morgan's law. So this time there are two predicates A and B. Here are their truth values. And here are two formulas. Remember this is a negation. Please write down the truth values of these two formulas in these boxes.

24 - Exercise de Morgans Law

[Click here to watch the video](#)

de Morgan's Law			
A	B	$\neg(A \wedge B)$	$\neg A \vee \neg B$
True	True	False	False
True	False	True	True
False	True	True	True
False	False	True	True

Figure 431: Exercise de Morgans Law

Truth of Implications		
A	B	$A \Rightarrow B$
True	True	True
True	False	False
False	True	True
False	False	True

Figure 432: Truth of Implications

That's good David. So the de Morgan's law is saying that when we try to distribute negation over the predicate inside the parentheses that are connected with a conjunction, then the conjunction becomes a disjunction between the negations of the pre, predicates. The same would have been true if we had a disjunction here. When we distribute the negation, it would have become a conjunction here. David, before we go ahead further, let's remember why we are trying to do all of this. So do you recall we said in the beginning of the lesson that a logical agent will have a knowledge base, and then formal rules of inference that will apply on these sentences as knowledge base. The knowledge base itself may be coming from many places. Some sentences in the knowledge base may be boot strapped into the logical agent. Other sentences may be coming from perception. Now when we're trying to apply these rules of inferences to the synthesis of the knowledge base it is sometimes very useful to rewrite the sentences in different forms. And that's what we are trying to do. These properties will allow us to rewrite the sentences in such a way that we can in fact apply the rules of inferences that we will see in a minute.

So it can be a little bit weird to talk about the truth value of an implication sentence. What we're really saying here is, whether or not this implication actually holds true. So let's take three different implications to see this. First let's think of the implication, feathers implies bird. All birds have feathers and only birds have feathers. So, we know that if an animal has feathers, then it is a bird. That's true. On the other hand, let's take the implication, if scales then bird. Lots of animals with scales aren't birds and in fact no animals with scales are birds. So the implication, scales implies birds. Would be False. For our third example, let's take the implication, flight implies bird. If we have a penguin, flight is False. But the penguin is still a bird. So, flight can be false and bird can still be true, meaning the implication can still be true here. On the other hand, if we have a cat, flight if False. And bird is False. So, the implication can still be true. So in this case, if flight was false, we can't actually make a determination on whether or not the animal is a bird.

26 - Implication Elimination

[Click here to watch the video](#)

Implication Elimination

Given:
 $a \Rightarrow b$

Rewrite as:
 $\neg a \vee b$

Given:
 $\text{Feathers} \Rightarrow \text{Bird}$

Rewrite as:
 $\neg \text{Feathers} \vee \text{Bird}$

25 - Truth of Implications

[Click here to watch the video](#)

Figure 433: Implication Elimination

As we go ahead and start applying rules of inferences to sentences in a knowledge base. We'll find it convenient to rewrite the sentences in a knowledge base. And sometimes it will be very useful to rewrite these sentences in the knowledge base in a manner that eliminates the implications in a sentence. And this is how we can eliminate the implication. If a implies b , than we can rewrite it as not a or b . We know this because the truth value of a implies b is exactly the same as your truth value of not a or b . We can take an example here. Supposing that we are given feathers imply bird. Then we can rewrite this as not feathers or bird. And intuitively, you can see the truth value of this. It is either the animal does not have feathers or, it is a bird. In a little bit, we will see that this is an important rewrite rule in doing certain kinds of logical proofs.

27 - Rules of Inference

[Click here to watch the video](#)

Rules of Inference: Instantiate general rules to prove specific claims.	
Modus Ponens	Modus Tollens
Sentence 1: $p \Rightarrow q$	Sentence 1: $p \Rightarrow q$
Sentence 2: p	Sentence 2: $\neg q$
\therefore Sentence 3: q	\therefore Sentence 3: $\neg p$
Feathers \Rightarrow Bird	Feathers \Rightarrow Bird
Feathers	\neg Bird
\therefore Bird	$\therefore \neg$ Feathers

Figure 434: Rules of Inference

Okay, now that we have looked at how to write sentences in the language of logic and also looked at how to rewrite the sentences, for example by eliminating implication, let us now look at what kinds of rules of inference can be implied and how can we apply them. One rule of inference is called Modus Ponens, and many of you may already be familiar with it. If I'm given a sentence s_1 which says p implies q , and another sentence s_2 which says p , then I could infer q from it. p implies q and p , therefore q , this symbol stands for therefore. Let's take an example, so imagine that I'm given that feathers imply bird. And I'm also given that feathers is true. Then, I can infer that bird must be

true. Now we can connect this to a logic agent. Imagine that there is a robot and I bootstrap that robot with the knowledge that feathers imply bird. Now the robot goes to a new region in the country and finds some animal which has feathers. The robot cannot conclude that that particular animal is a bird. So the first sentence came from something that I had bootstrapped into the knowledge of the robot. The second sentence came from the percepts of the robot. And the third sentence came from its logical inferencing. And this is how the robot can, in fact, go about making sound, complete inferences that are guaranteed to be correct. Here is a second rule of inference, this is called Modus Tollens. So again I have sentence s_1 , p implies q , and I have a second sentence, not q . And therefore, I can infer that not p . So let us take an example of Modus Tollens. Imagine that there is a robot that has been programmed, bootstrapped with the knowledge feathers imply bird. So that's part of its knowledge base already. This robot goes to a new country and is talking to the people in that country, and the people tell the robot a story about an animal that is not a bird. Therefore the robot may infer that that animal must not have feathers. So this is coming from the knowledge that is bootstrapped. This is coming from the new percept from the story. And this is coming from the logical inference. And once again, the logical inference is guaranteed to be sound and complete. You may already be familiar with this line of reasoning because this is another way of phrasing a contrapositive, that we see in other areas of logic.

28 - Prove Harry is a bird

[Click here to watch the video](#)

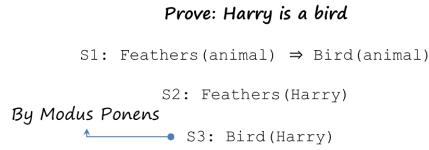


Figure 435: Prove Harry is a bird

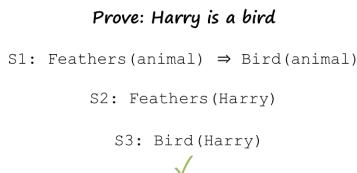


Figure 436: Prove Harry is a bird

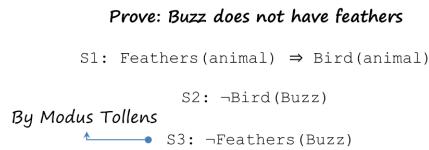


Figure 437: Prove Harry is a bird

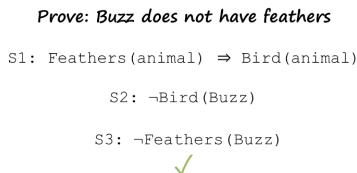


Figure 438: Prove Harry is a bird

Now you can see how we apply these rules of inferences on sentences in a knowledge base or philosophical agent to prove all kinds of sentences. See, imagine that an AI agent begins

with the knowledge that if an animal has feathers, it implies that the animal is a bird. Now it comes across Harry, who does have feathers. By Modus Ponens, therefore the AI agent can conclude that Harry is a bird. This completes the proof for our original goal of proving that Harry is a bird. Now let us suppose that a goal is to prove that Buzz does not have feathers. Once again, imagine an AI agent which begins with the knowledge that if an animal has feathers, it implies that the animal has, is a bird. The agent comes across a animal, which is not a bird. Then by Modus Tollens it can infer that buzz must not have feathers. This completes the proof for of a original goal of proving that buzz does not have feathers. Okay. So now, we have looked at two ways of proving the truth value of various sentences. The first way was just through truth tables. I could have sentences and logic. Then I could write another sentence. And ask myself, what, what is the truth value of this sentence? I could construct a truth table for that sentence, composed of the truth values of all the predicates, with some of which might be coming from earlier sentences. The second way in which we have seen how we can prove the truth values of sentences and logic is by applying these rules of inferences like modus ponens and modus tollens. This is very powerful, and in fact the power of this logic has been known since before the birth of AI. As computer scientists however, we'll analyze this power in a slightly different way. Yes, we can use method of truth tables to construct a truth table for any arbitrary sentence. However, the sentence was complicated. Then the truth table very soon will become very complex. Computationally, that is infeasible for very long, large sentences. Similarly, yes we can apply simply modus ponens and modus tollens to find the truth value of many sentences. But if the knowledge base consisted of a very large number of sentences, instead of just one or two sentences, then the kinds of inferences, number of inferences I can draw from those sentences simply by applying modus ponens and modus tollens, will be very large. Or if I had to find the truth value of a single sentence, then the different pathways I could take in order to get to the truth value of

those sentences can make for long, large problem space. So while these methods of proving the truth with your sentences and logic have been around for a long time. These methods are not computationally feasible. At least not for complex tasks. At least not for agents that have only limited computational resources and from who we want near realtime performance

29 - Universal Quantifiers

[Click here to watch the video](#)

For one animal:

$$\text{Lays-eggs(animal) \wedge Flies(animal) \Rightarrow Bird(animal)}$$

For all animals:

$$\forall x[\text{Lays-eggs}(x) \wedge \text{Flies}(x) \Rightarrow \text{Bird}(x)]$$

"Universal Quantifier"

Figure 439: Universal Quantifiers

For one animal:

$$\text{Lays-eggs(animal) \wedge Flies(animal) \Rightarrow Bird(animal)}$$

For at least one animal:

$$\exists y[\text{Lays-eggs}(y) \wedge \text{Flies}(y) \Rightarrow \text{Bird}(y)]$$

"Existential Quantifier"

Figure 440: Universal Quantifiers

Before we show you, a computationally more feasible way of proving theorems in logic, or proving the truth value of sentence in logic. We should point out that so far, we have been using only propositional logic. Propositional logic is sometimes also called the zero-if order logic. The key aspect of propositional logic, is that it does not have any variables. So as an example, I may have a sentence that says if the animal Lays-eggs, and the animal Flies, then the animal is a Bird. And here I'm talking about a specific animal. Well, sometimes I might want to talk about, animals in general, any animal, all animals. In that case, I would want to introduce a variables in it. So

in first audilogic, otherwise known as predicate calculus, I might want to say something like. If x Lays-eggs and x Flies, then x is a Bird. Which has a set form very similar to form here, except that instead of animal, I now have a variable. Now, I have a variable here. But, I must also specify the range of the variable. And what I really want to say here is for all animals. Therefore I'll introduce a new quantifier over the variable x . This quantifier is called Universal Quantifier. It is denoted with the symbol, this is the symbol for Universal Quantifier. And this says now for all x , if x Lays-eggs, and x Flies it implies that x is a Bird. One thing to note here is that, I could have rewritten this sentence, with the Universal Quantifier back into proposition logic. But, having lots of sentences like this. In proposition logic. So I could've said Lays-eggs (animal) one, Flies (animal) one implies Bird (animal) one, Lays-eggs (animal) two, and Flies (animal) two implies Bird (animal) two. And so on and so forth, for each and every animal that is possible. But, by writing it in the form of a variable, a Universal Quantifier statement, I can reduce the number of sentences I have to write into just one sentence. So we have introduced variables, and we have talked at least about one quantifier so far, the Universal Quantifier, that applies for all values that that variable can take. Sometimes I might want to specify a different range of the variable. Not all values of the variable can take, but, at least some values of the variable I can take. So consider again, this sentence, here the animal is [referring] a specific animal. Now let's look at the second sentence on this screen. And this sentence is the variable y . It says if y Lays-eggs and y Flies then it implies that y is a Bird. This sentence is a very similar form, to the previous except for the variable y . I can specify the value, that the variable y can take. This time I want to specify not that this sentence is true for all values of y , for all animals, but simply that it is true for. Some at least one animal in which case I'll use an Existential Quantifier. Here is the symbol for an Existential Quantifier, this Existential Quantifier says that there is at least one animal, for which this sentence happens to be true.

30 - A Simple Proof

Click here to watch the video

We know:
S1: $\neg\text{can-move} \Rightarrow \neg\text{liftable}$

We find:
S2: $\neg\text{can-move}$

How do we prove the box is not liftable?

Figure 441: A Simple Proof

S1: $\text{can-move} \vee \text{liftable}$
S2: $\neg\text{can-move}$
S3: liftable

How do we prove the box is not liftable?

Figure 445: A Simple Proof

We know:
S1: $\neg\text{can-move} \Rightarrow \text{liftable}$

By implication elimination:
S1: $\text{can-move} \vee \neg\text{liftable}$

We find:
S2: $\neg\text{can-move}$

We assume:
S3: liftable

How do we prove the box is not liftable?

Figure 442: A Simple Proof

How do we prove the box is not liftable?

Figure 446: A Simple Proof

S1: $\text{can-move} \vee \text{liftable}$
S2: $\neg\text{can-move}$
S3: liftable

How do we prove the box is not liftable?

Figure 443: A Simple Proof

S1: $\text{can-move} \vee \text{liftable}$
S2: $\neg\text{can-move}$
S3: liftable

How do we prove the box is not liftable?

Figure 444: A Simple Proof

Okay, let us set aside predicate calculus, and return back to population logic. Recall that we had found ways of writing sentences in population chronologic. We had found rules of inferences, we could prove theorems. We could find the truth value of new sentences. However, we found that those methods were computationally, not very efficient. So AI has developed more efficient methods. One of those methods is called Resolution Theorem Proving. Let us take an example to illustrate how resolution theorem proving works. So, imagine there is a robot, and this robot. Is working on an assembly line, it's a factory robot, and on the assembly line are coming weird kind of widgets. The robot's task is to pick up each widget, as it comes on the assembly line and put it in a truck. However, there are some humans in this factory. Who play a joke on the robot once in a while, they glued the widget to the assembly line belt, so that, when the robot tries to move it, it can not move it. But the robot is a smart robot, this is a logical agent, so when it can not move it. It uses its logical reasoning, to figure out that the boxes aren't liftable. And the moment it knows that the boxes aren't liftable, it lets go of the box and moves onto the next one.

Everyone got the story? All right. So let us suppose that the robot begins with some knowledge in its knowledge base. And this knowledge in its knowledge base, that it begins with says that if cannot move, then it implies that not liftable. Now, it tries to move the box, the next box in the widget. It's biceps tells it, it can not move. It needs to prove that it's not liftable. And of course this is a preview example and I'm sure you'll understand it. You can put essentially a class of the modest components to prove that it's not liftable. If p then q , p therefore you can infer q . But, we'll use this example to show. How does resolution theorem proving works? So, the first step in resolution theorem proving is, to convert every sentence into a conjunctive normal form. A conjunctive normal form of a sentence, can have one of three conditions. It can have a literal. That can be either a positive atom, or a negative atom. It can have this disjunctional literals like here can-move, or not liftable, or it can have a conjunction of disjunctional request. In this example the third condition doesn't occur. So, the first thing we must do is to take the first sentence. The negation of not move implies not liftable. And remove the implication, because an implication cannot occur in conjunctive normal form. So the first thing we need to do is, to rewrite the sentence, the first sentence, to remove the implication. Because the implication cannot occur in a conjunctive normal form. So now we use the. Implication elimination rewrite rule. To rewrite this in the form of can-move, or not liftable. Remember that was alpha implies beta becomes, not alpha or beta. So the not of negation of can-move becomes can-move or not liftable. So, we have done it for the first sentence. This is now in a conjunctive normal form. We can do the same thing for the second sentence, but wait, the second sentence already is in a conjunctive form. We don't have to do anything. Now, the robot wants to prove that their box is not liftable. Resolution to improving, is like proof by refutation. To do proof by refutation we will take the negation of what we want to prove. We wanted to prove not liftable would take its negation, which makes it liftable. Okay, so now we got three sentences. This one's

the first sentence that the robot was bootstrap with, you've just converted to a conductor normal form. This was the sentence that came from a it saw that the box cannot move. And this is the sentence throughout the negation of the sentence, the refutation of the sentence that it wants to prove. So we have three sentences now. The first sentence came from the bootstrapping, of the robot's knowledge base. This is the axiom that the robot assumes to be true. The second sentence came from its percepts. The robot tried to move the box, it could not move it. The third sentence is coming from taking the negation of what the robot wants to prove. It wants to prove it's not liftable. So, it's going to take this negation of it and then, sure that it's going to lead to a null condition that we'll view as a contradiction. The resolution for improving lawless begin with a liftable in the sentence that we want to prove. So here that sentence is liftable, and we'll look for a sentence that contains a negation of liftable in this sentence that we want to prove. So the sentence here was liftable, sentence S1 contains liftable which is a negation of that so we pick S1 and not S2. Note, how efficient it was to decide what sentence on the knowledge based to go to. In sentence container negation of the liftable. So, liftable and not liftable can not both be true. We know that, and therefore we can eliminate them. This is called resolution. We resolve unliftable and we remove them from the sentences. Now, we were sentence as S1, that leaves us can move. So, now we pick a sentence, that has the negation of literal can-move. Sentence S2 has a negation of that, and we can resolve one can move, they can not both be true. When we resolve on both of them, those get eliminated as well. And now we see we've reached another condition. This null condition represents a contradiction, and now we can infer that liftable cannot be true, therefore not liftable is true. The robot has proved not liftable. And in this case it appears as a resolution theorem improving is more complex there's more respondents. In general it is not. It just appears here, because this condition happened to fulfil the form of more respondents directly. In general, deciding on which sentence to apply the modest ponents on, and

how to combine those groups of inferences don't suddenly become [turns out to be computationally] harder than deciding how to apply the resolution and improvements.

S1: can-move $\vee \neg$ liftable $\vee \neg$ battery-full
 S2: \neg can-move
 S3: battery-full
 S4: liftable

31 - A More Complex Proof
[Click here to watch the video](#)

How do we prove the box is not liftable?

Figure 450: A More Complex Proof

We know:
 S1: \neg can-move \wedge battery-full $\Rightarrow \neg$ liftable

We find:
 S2: \neg can-move
 S3: battery-full

How do we prove the box is not liftable?

S1: can-move \vee liftable $\vee \neg$ battery-full
 S2: \neg can-move
 S3: battery-full
 S4: liftable

How do we prove the box is not liftable?

Figure 447: A More Complex Proof

Figure 451: A More Complex Proof

We know:
 S1: ~~can-move \wedge battery-full \Rightarrow liftable~~

By implication elimination:
 S1: $\neg(\neg$ can-move \wedge battery-full) $\vee \neg$ liftable

We find:
 S2: \neg can-move
 S3: battery-full

How do we prove the box is not liftable?

S1: can-move \vee liftable $\vee \neg$ battery-full
 S2: \neg can-move
 S3: battery-full
 S4: liftable

How do we prove the box is not liftable?

Figure 448: A More Complex Proof

Figure 452: A More Complex Proof

We know:
 S1: ~~can-move \wedge battery-full \Rightarrow liftable~~

By implication elimination:
 S1: $\neg(\neg$ can-move \wedge battery-full) $\vee \neg$ liftable

By deMorgan's Law:
 S1: can-move $\vee \neg$ battery-full $\vee \neg$ liftable

We find:
 S2: \neg can-move
 S3: battery-full

How do we prove the box is not liftable?

S1: can-move \vee liftable $\vee \neg$ battery-full
 S2: \neg can-move
 S3: ~~battery-full~~
 S4: liftable

How do we prove the box is not liftable?

Figure 449: A More Complex Proof

Figure 453: A More Complex Proof

S1: can-move \vee liftable \vee battery full
S2: \neg can-move
S3: battery full
S4: liftable

How do we prove the box is not liftable?

Figure 454: A More Complex Proof

S1: ~~can-move~~ \vee liftable \vee battery full
S2: ~~can-move~~
S3: battery full
S4: liftable

How do we prove the box is not liftable?

Figure 455: A More Complex Proof

S1: can move \vee liftable \vee battery full
S2: ~~can move~~
S3: battery full
S4: liftable

How do we prove the box is not liftable?

Figure 456: A More Complex Proof

Let us make this example a little bit more complicated. Complicated enough that it cannot be proven simply by applying one instance of modus ponens. Imagine that a robot proved to itself that this box is not liftable. And the humans in the factory who were trying to make fun of the robots said to the robot well, the reason it's not liftable is not because it's not movable, but because your battery is not working. So now the situation is more complicated, the robot must also check its battery. So now the robot begins with slightly different knowledge in this knowledge case. So suppose that the knowledge in its knowledge basis, cannot move and battery

full means it's not liftable. It finds from its concept. Again, it cannot move, so it checks its battery, and it checks its battery and it finds that the battery's full. So then two new sentences that get written in the knowledge base. By the knowledge base contains three sentences. As earlier, the resolutions you're improving, the agent must convert all these sentences, in its knowledge base into a conjunctive normal form. That means that the sentences can contain a literal or a disjunctive literal or a conjunction of disjunctive literals. So if we begin by removing the implication from sentence one, because an implication cannot occur in a conjunctive normal form. So when we remove implication from the first sentence we get this sentence. Where the sentence is not yet satisfactory, it is not yet a conjunctive normal form because this is a disjunction of conjunctions. And what we want are conjunctions of disjunctions. So we apply the deMorgan's Law and now we get the following sentence. We're simply taking the negation inside which flips the conjunction into a disjunction and now we have three liftables connected with disjunctions and this is a conjunctive normal form, disjunctional liftable. So now we have in the knowledge base, three sentences, all three of them in the conductor normal form, either literals or disjunctive literals. Recall that the robot wanted to prove not liftable. So it takes the negation of that, this is again proof by refutation, so it considers liftable. So now this knowledge base has four sentences. These four sentences coming from the negation of what it wants to prove. Once again the reasoning begins by the literal that it wants to prove, in this case liftable. It finds a sentence in which the negation of literal is true. So once again, we begin with the sentence, S4 because that is what we want to prove. And we find a sentence in the knowledge base which contains a literal which is a negation of the literal in this sentence S4 that we want to prove. We resolve on this because they both cannot be true, and resolution here simply means that we drop them. Now, in the sentence S1 that is under consideration currently, we have two literals. We can begin with either one of them. Let us begin with not battery full. We'll try to find

a sentence which contains a negation of this particle electrode. There is a sentence, S3, which is a negation of this so we resolve on this. Battery full and not battery full because they both cannot be true. We'll drop them. Now in sentence S1 we are left with just one literal, can-move. We can try to find a sentence in the knowledge base which contains a negational judge literal. Here it is, and so we can resolve on them. And we resolve on them, we drop them. And once we drop them, then we have a null condition, which stands for a contradiction. So we reached a contradiction, therefore the assumption that this was liftable cannot be true, therefore not liftable is true, and we have just shown that resolution theorem proving in this case proves what the robot wanted to prove. One important point to note here is the issue of focus of attention. Often when the problem space is very complex, for example, when the number of sentences is really large, the sentences are very complex, it can become really hard for the logical agent to decide what to focus on. But because we have converted everything in a conjunctive normal form, and because resolution theorem proving is making use of resolution, therefore at any particular time, the logical agent knows exactly what to focus on. You always begin with this literal, you always try to find a sentence which contains this negation. You always resolve on that. You take the remaining literal in the sentence and proceed forward. This focus of attention, this computational efficiency of resolution theorem proving is arising because a [of what are called] called horn clauses. A horn clause, is a disjunction that contains at most one positive literal. This is happening in S1. This is a disjunction that contains at most one positive recall. This is a negative recall, this is a negative recall, and the fact that it contains just one positive recall, is a very powerful idea because that's where the focus of attention comes from.



If an animal has wings and does not have fur, it is a bird.

[Write in formal logic:](#)

(Use the predicates has-wings, has-fur, and bird)

How do we prove this is a bird?

Figure 457: Exercise Proof I

Let us do an exercise together to make sure that we get rich solutions for improving. So consider this sentence if an animal has wings and does not have fur, it is a bird. Write this sentence down in formal logic. You can use the predicates, has-wings, has-fur, and bird.

33 - Exercise Proof I

[Click here to watch the video](#)



If an animal has wings and does not have fur, it is a bird.

[Write in formal logic:](#)

(Use the predicates has-wings, has-fur, and bird)

How do we prove this is a bird?

Figure 458: Exercise Proof I

David, what did you write? So starting at the beginning, has wings becomes the predicate has wings. We're doing a conjunction so and, does not have fur, so not has fur, those two things imply that it's a bird. That's good, David. Now, let us put this in a form, in a conjunctive normal form that we can use in resolution theorem proving.

32 - Exercise Proof I

[Click here to watch the video](#)

34 - Exercise Proof II

[Click here to watch the video](#)



$\text{has-wings} \wedge \neg\text{has-fur} \Rightarrow \text{bird}$
 Use implication elimination to rewrite as a conditional:
 (Use the predicates has-wings, has-fur, and bird)

How do we prove this is a bird?

Figure 459: Exercise Proof II



$\neg(\text{has-wings} \wedge \neg\text{has-fur}) \vee \text{bird}$
 Use de Morgan's Law to rewrite in conjunctive normal form:
 (Use the predicates has-wings, has-fur, and bird)

How do we prove this is a bird?

Figure 461: Exercise Proof III

35 - Exercise Proof II

[Click here to watch the video](#)



$\text{has-wings} \wedge \neg\text{has-fur} \Rightarrow \text{bird}$
 Use implication elimination to rewrite as a conditional:
 $\neg(\text{has-wings} \wedge \neg\text{has-fur}) \vee \text{bird}$
 (Use the predicates has-wings, has-fur, and bird)

How do we prove this is a bird?

Figure 460: Exercise Proof II



$\neg(\text{has-wings} \wedge \neg\text{has-fur}) \vee \text{bird}$
 Use de Morgan's Law to rewrite in conjunctive normal form:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 (Use the predicates has-wings, has-fur, and bird)

How do we prove this is a bird?

Figure 462: Exercise Proof III

What did you get David? So this is a little bit more complicated. We know that from our earlier formula if a implies b, then to rewrite it with implicational elimination we write, not a or b. So or b is pretty straightforward, or bird. We take the not of the conjunction over here and say not has wings and not has fur. Now some of you may have jumped straight to writing this in full conjunctive normal form, but now we're going to move on and do that last step.

36 - Exercise Proof III

[Click here to watch the video](#)

38 - Exercise Proof IV

[Click here to watch the video](#)



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$

What sentence would be assumed
to facilitate the proof?
S4:

How do we prove this is a bird?

Figure 463: Exercise Proof IV



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$
 S4: $\neg\text{bird}$

What part of S1 would we
eliminate first?
 $\neg\text{has-wings}$
 has-fur
 bird

How do we prove this is a bird?

Figure 465: Exercise Proof V

So imagine that your robot has this in its knowledge base and goes into a country where it finds an animal and from its perspective knows that this animal has wings and does not have fur. So these two additional centers is gotten in the knowledge base. Now the robot wants to prove that this is a bird. In order to [do resolution theorem] proving, what should the robot begin by writing in this box? What should the S4 sentence be?

39 - Exercise Proof IV

[Click here to watch the video](#)



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$

What sentence would be assumed
to facilitate the proof?
S4:

How do we prove this is a bird?

Figure 464: Exercise Proof IV



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$
 S4: $\neg\text{bird}$

What part of S1 would we
eliminate first?
 $\neg\text{has-wings}$
 has-fur
 bird

How do we prove this is a bird?

Figure 466: Exercise Proof V



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$
 S4: $\neg\text{bird}$

What part of S1 would we
eliminate first?
 $\neg\text{has-wings}$
 has-fur
 bird

How do we prove this is a bird?

Figure 467: Exercise Proof V

What do you think, David? So in resolution theorem proving, we always assume the opposite of what we're trying to prove. We're doing proof by refutation. So we're trying to prove that it is a bird, so we're going to assume that it's not a bird. That's right.

40 - Exercise Proof V

[Click here to watch the video](#)

What do you think David? So in resolution theorem proving, we start with whatever it is we assumed and look for the negation of that in an earlier sentence. Here we find not bird, and bird in S1. So we're going to resolve on not bird and bird and leave both of those out. That's good.

42 - Exercise Proof VI

[Click here to watch the video](#)



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$
 S4: $\neg\text{bird}$

What do we do next?
 o Resolve on S2 and $\neg\text{has-wings}$ from S1
 o Resolve on S2 and has-fur from S1
 o Resolve on S3 and $\neg\text{has-wings}$ from S1
 o Resolve on S3 and has-fur from S1

How do we prove this is a bird?

Figure 468: Exercise Proof VI

So what shall we do next? What should we resolve on next?

43 - Exercise Proof VI
[Click here to watch the video](#)



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$
 S4: $\neg\text{bird}$

What do we do next?
 o Resolve on S2 and $\neg\text{has-wings}$ from S1
 o Resolve on S2 and has-fur from S1
 o Resolve on S3 and $\neg\text{has-wings}$ from S1
 o Resolve on S3 and has-fur from S1

How do we prove this is a bird?

Figure 469: Exercise Proof VI



S1:
 ~~$\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$~~
 S2: ~~has-wings~~
 S3: ~~has-fur~~
 S4: ~~$\neg\text{bird}$~~

How do we prove this is a bird?

Figure 470: Exercise Proof VI



S1:
 $\neg\text{has-wings} \vee \text{has-fur} \vee \text{bird}$
 S2: has-wings
 S3: $\neg\text{has-fur}$
 S4: $\neg\text{bird}$

How do we prove this is a bird?

Figure 471: Exercise Proof VI

Out of four choices, which one did you pick, David? So, I picked the first one, but I think there's actually two correct answers. What we're looking for is something else in S1 that has a negation in another sentence. Not has wings has a negation in S2, so we could resolve on S2 with the not has wings portion of S1. Has fur also has a negation in S3. So we could also resolve on S3 and the has fur portion of S1. This is right, David. At the end of this, we're left with null, which is a contradiction. Therefore, an assumption that is not bird is false. Therefore, it must be a bird and the robot has just proved that this must be a bird. Note what we have done. We have mechanized parts of logic. And sort of coming up with large truth tables. And it's sort of coming up with complex chains of inference based on modus ponens and modus tollens. We have found in our garden resolutions are improving, which is an efficient way of proving sentences and the truth values. This is how it works. Take all the sentence and the knowledge base, convert them into conjecture novel form. Take what you want to prove and its negation. Put that as a new sentence. Now, starting with this particular sentence, a new sentence. Find the literal in another sentence on which you can resolve, and keep on doing it until you find a null condition, which is a contradiction. If you don't find a null condition, if you don't find a contradiction, that means that what you started with cannot be proved.

44 - Assignment Logic
[Click here to watch the video](#)

Assignment

How would you represent
 Raven's progressive matrices
 using formal logic?

Figure 472: Assignment Logic

So how would you use formal logic to develop an agent that can solve Raven's progressive matrices? As with production systems, we can kind of face this at two different levels. One, you could use formal logic to represent the overall algorithm the agent uses to solve any new problem. Or secondly, the agent could use formal logic to develop its understanding of a new problem that just came in. It could then use those formal rules to develop the transformations that occur within the problem and transfer those transformations into the new answer. Alternatively, you could also use formal logic to allow your agent to prove why it's answer to a particular problem is correct. Then if the answer is actually incorrect, the agent may have the information necessary to go back and repair it's reasoning and do better next time.

45 - Wrap Up

[Click here to watch the video](#)

To recap...

- Formal notation
- Properties of truth values
- Rules of inference
- Proof by refutation

Figure 473: Wrap Up

So today, we've talked about formal logic in order to set up a kind of formal language for us to reason with going forward. We started off by talking about formal notation, including conjunctions and disjunctions, so that we can write sentences in formal logic. We use that to then talk about truth tables, and exam some of the properties that we need going forward, like De Morgan's law. Using that, we investigated some of the inferences that we can draw using formal logic. And finally, we looked at proof by reputation which kind of capitalizes on everything we've talked about so far. Next time, we'll be discussing planning, which leverages the formal

logic that we've developed in this lesson. It allows agents to reason more formally about initial and goal states. Interestingly, planning actually has its history in the kinds of proofs we've developed here. Originally, agents would prove that a particular plan would work, so that's why we talk about formal logic before we talk about planning.

46 - The Cognitive Connection

[Click here to watch the video](#)

The connection between logic and human cognition is interesting. Logic is a very important school of thought in AI, for several reasons. One reason is that logic provides a very formal and precise way of reasoning. Another reason is that logic provides a formal notation for expressing how intelligent agents reason, whether or not they're using logical reasoning. But does this mean that logic is also the basis of cognition? It suddenly appears that humans use logic some of the time. For example, I may have a statement like if I get a big bonus, I'll take a long vacation. I did get a big bonus, therefore I may infer I'll take a long vacation. This is clearly. But simply because of a behavior appears to be logical, does not necessarily imply that we use logic as a fundamental reasoning strategy. I might solve a new problem, by analogy to a previously encountered problem, I did not use logic, but my behavior appears to be logical. The logic that we have considered so far is deductive logic. But a lot of human reasoning is inductive or abductive in its character. If you haven't come across an abductive so far, we will discuss in detail later in the class. Deduction has to do with reasonings from causes to effects. Abduction has to do with reasoning from effects to causes. When you go to a doctor, you go with some signs and symptoms. Those are the effects. The doctor comes up with a disease category. That's the cause. Abduction is reasoning from data to an explanation to a disease category for the data. Induction is given some relationship between cause and effect for a sample. How do we generalize it between a cause and effect relationship for a population? So while human reasoning appears to be inductive and abductive in character much of the time,

the logic that we have considered so far is deductive. That's yet another issue that we'll return to later in the class.

47 - Final Quiz

[Click here to watch the video](#)

All right, write what you learned in this lesson in this box right here.

48 - Final Quiz

[Click here to watch the video](#)

Thank you for filling out this box. It helps us understand how the learning is going in the class.

Summary

Logic provide the framework for formal notation/language for reasoning and inferences. In other words, logic provides a formal and precise way of reasoning. It also allows agents to reason more formally about initial and goals states and helps in planning. Deduction is term used for reasoning from causes to effects; Abduction is the term used for reasoning from effects to causes; and Induction is generating a generic rule, given the cause and its effect.

References

1. Winston P., Artificial Intelligence,

Pages 283-303.

Optional Reading:

1. Winston Chapter 13; [Click here](#)

Exercises

Exercise 1:

It is said that a cat attempts to pass through a narrow space (such as the bars on a window) only if the width of space is more than the size of its whiskers. Now consider a robot cat, which doesn't have any whiskers. The robot cat attempts to pass through the window bars if the coast is clear, e.g., there is no dog around. Sometimes however the robot cat gets trapped between bars as it attempts to pass through. So the robot cat has the following propositions: COAST_CLEAR, BARS_PASSABLE, and TRAPPED. Write the fact that if the coast is clear and the bars are passable, then the robot will not get trapped as a logical implication. Now when the robot cat gets trapped between the bars, show how it can use resolution to figure out that the bars are not passable after all.

Lesson 13 - Planning



A man who does not plan long ahead will find trouble right at his door.
— Confucius.

Failing to plan is planning to fail.
— Effie Jones.

01 - Preview

[Click here to watch the video](#)

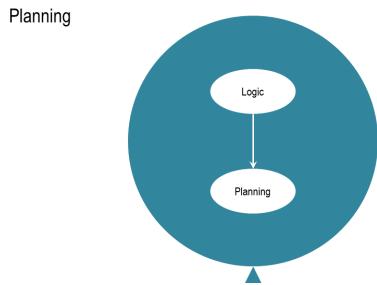


Figure 474: Preview

- Lesson Preview**
- States, goals, and operators
 - Conflicts in planning
 - Partial-order planning
 - Hierarchical task networks

syntax we learned from logic last week to set up the specification of goals and states, and operators, for moving between states and achieving goals. We'll see that when there are multiple goals, there can be conflicts between them. We'll describe a specific technique called partial-order planning that avoids conflict between multiple goals. Finally, we'll talk about a representation called hierarchical task networks that allows us to make complex hierarchical plans.

02 - Block Problem Revisited

[Click here to watch the video](#)



Figure 476: Block Problem Revisited

Figure 475: Preview

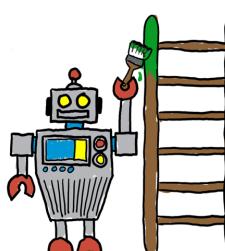
Today we'll talk about planning. Recall that we had said an intelligent agent maps perceptual histories into actions. Planning is a powerful method for action selection. We'll use the

In order to look at planning in detail, let us consider this problem that we have encountered earlier. This is a blocks world, in which there is a robot which has to move the blocks from the current state, to the goal state. The robot has only

one arm, so it can move only one block at a time. It can move only a block which does not have some other block on top of it. Earlier, when we considered this method, we had looked at weak AI methods like means-ends analysis. Now we're going to look at more systematic, knowledge-based methods. One question unanswered in our previous discussion was, how can we find which goal to select, among the various goals that are here? They simply said, the agent might select a goal. Now we will look at how the agent can in fact do the goal composition and select the right goal to pursue?

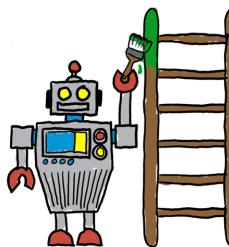
03 - Painting a Ceiling

[Click here to watch the video](#)



Goals:
The ceiling is painted
and the ladder is
painted

In propositional logic:
 $\text{Painted}(\text{Ceiling})$



Goals:
The ceiling is painted
and the ladder is
painted

In propositional logic:
 $\text{Painted}(\text{Ceiling})$

Goal State:

Figure 477: Painting a Ceiling

It is considered a little bit more realistic problem. Imagine that you were to hire a robot, and a task of the robot was to paint the ceiling in your room, and also paint the ladder. So, two goals here, paint the ladder paint the ceiling and note that the two goals are in conflict. Because if the robot paints the ladder first, the ladder will become wet. And this robot cannot climb on it, in order to paint the ceiling. So, the robot must really first paint the ceiling, then climb down, then paint the ladder and everything is okay. You would expect a human to get this almost immediately, you probably got it almost immediately. You have to paint the ceiling first, before you paint the ladder. Of course, every time I've hired construction workers, they always paint the ladder first. Then they go and take a break, and so that I had to pay them anyway. We'll accept that kind of behavior from

human construction workers, we would not accept that from robot construction workers. The robots must be intelligent, they must know how to prioritize the goals. Well, in order to reason about these goals, the robot first must be able to represent them. So, how can we present the goal of painted ceiling? Now that we have learned about propositional logic, here is a preposition that can represent painted ceiling. This is the object, this is the predicate on that.

04 - Exercise Goals

[Click here to watch the video](#)

Figure 478: Exercise Goals

Goals:
The ceiling is painted
and the ladder is
painted

How would we
represent second part
of the goal state?

In propositional logic:
 $\text{Painted}(\text{Ceiling})$

How would we
represent the goal
state as a conjunction?

Goal State:

Figure 479: Exercise Goals

So in this box, please write down the second goal of painting the ladder in propositional form. And having done that, in this box, write down how would we represent the two goals as a conjunction.

05 - Exercise Goals

[Click here to watch the video](#)

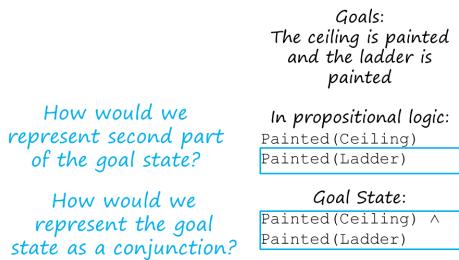
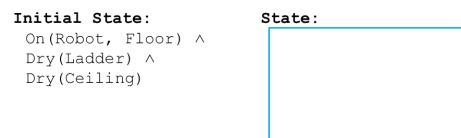


Figure 480: Exercise Goals

David, what do you think? So the second goal is pretty straightforward. We're using the same proposition, but here we're operating on the Ladder instead of on the Ceiling. And then for the goal state we're just going to link the two together with a conjunction, so we're saying that our goal state is Painted(Ceiling) and Painted(Ladder). Good. Let's move on then.

06 - States

[Click here to watch the video](#)



How would we represent a state where the robot
is on the ladder and the ceiling is painted?

Figure 482: Exercise States

Now that we have learned to specify the initial state of the world and the ghosts of the world, let us do an exercise in specifying other states of the world. So please write down in this box, the state of the world that would occur after the robot is on the ladder and the ceiling has been painted.

08 - Exercise States
[Click here to watch the video](#)

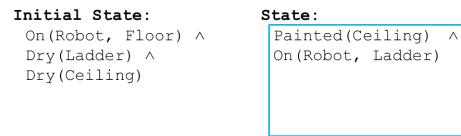


Figure 481: States

So, we just talked about goals and the goal state of the goals. In order to specify the goal fully, we need to not only specify the goal state, but also to specify the initial state. So, let's do that. Let us suppose that the initial state in this world is, that the robot is on the floor. Note, how I'm writing this. On is a predicate, that says this is a two [tuple] and I'm reading this as Robot On Floor. So Robot on Floor, and the Ladder is Dry, and the Ceiling is Dry. So, this is the Initial state, this is the Goal State. Now we have fully specified the goal. Let's now ask, how can the robot come up with a plan?

07 - Exercise States

[Click here to watch the video](#)



How would we represent a state where the robot
is on the ladder and the ceiling is painted?

Figure 483: Exercise States

How did you specify the state, David? So, earlier when we were talking about our goal state, we had specified that we wanted our ceiling to be painted. And we wrote that as Painted (Ceiling). So if in this state the ceiling is painted, I'll write that as Painted (Ceiling). Earlier in the initial state, we said that the robot was on the floor. And now we know that the robot is on the ladder. So I'm going to say Robot On Ladder. And this is an and, so I'm going to join them with a conjunction right there. I wasn't giving any other information about the world, so I'm not including anything else in this state. That's good, David. In general, if there is information about the initial state, additional information for

example dry ladder and dry ceiling, then those associations about the world can be propagated to the subsequent states, provided that no operator in the middle of this initial state and this state actually negates or deletes any of those assertions. We'll see more of this in just a minute.

09 - Operators

[Click here to watch the video](#)

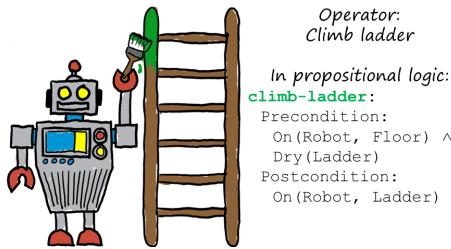


Figure 484: Operators

Now that we have learned how to specify states in planning, let us look at how to specify the operators. So consider the operator, climb-ladder. You might specify the climb-ladder operator, and any other operator in general, to a set of preconditions, and post-conditions. So for the climb-ladder operator, the precondition might be that the robot is on the floor and the ladder is dry. Notice that these are being written in the language of propositional logic. And the post-condition for the climb-ladder operator might be that the robot is on the ladder. This captures the notion that the robot has climbed the ladder. It was earlier on the floor and later it's on the ladder. Several things to note here. First, by convention the precondition does not have any negative leg holds. All the leg holds in the precondition are positive. Post conditions of operators may have negative leg holds. Second, the meaning of this pre-condition, and post conditions is that, these assertions are true in the world before this operator can be applied. And that these assertions become true in the world after this operator has been applied. This captures the syntax of the operator climb-ladder. The semantics of this operator is that this operator can be applied if and only if the preconditions of the operator

are true in the world. It cannot be applied if the preconditions are not true in the world

10 - Exercise Operators

[Click here to watch the video](#)

climb-ladder: Precondition: <code>On(Robot, Floor) ^ Dry(Ladder)</code> Postcondition: <code>On(Robot, Ladder)</code>	paint-ceiling: Precondition: <code>On(Robot, Ladder)</code> Postcondition: <code>Painted(Ceiling) ^ Dry(Ceiling)</code>
descend-ladder: Precondition: <code>On(Robot, Ladder)</code> Postcondition: <code>On(Robot, Floor)</code>	paint-ladder: Precondition: <code>On(Robot, Floor)</code> Postcondition: <code>Painted(Ladder) ^ Dry(Ladder)</code>

Figure 485: Exercise Operators

Now that you have learned how to specify an operator, such as climb ladder. Let us do some exercises about how to specify other operators, like descend ladder, paint ceiling and paint ladder. In these boxes, please write down the precondition and the post-condition in the same notation

11 - Exercise Operators

[Click here to watch the video](#)

climb-ladder: Precondition: <code>On(Robot, Floor) ^ Dry(Ladder)</code> Postcondition: <code>On(Robot, Ladder)</code>	paint-ceiling: Precondition: <code>On(Robot, Ladder)</code> Postcondition: <code>Painted(Ceiling) ^ Dry(Ceiling)</code>
descend-ladder: Precondition: <code>On(Robot, Ladder) ^ Dry(Ladder)</code> Postcondition: <code>On(Robot, Floor)</code>	paint-ladder: Precondition: <code>On(Robot, Floor)</code> Postcondition: <code>Painted(Ladder) ^ Dry(Ladder)</code>

Figure 486: Exercise Operators

David, what did you come up with? So the descend-ladder operator is pretty similar to the climb-ladder operator. We still have to have a dry ladder in order to climb up or down the ladder. This time in order to climb down the ladder, we have to already be on the ladder. So our pre-conditions are that we are on the ladder and the ladder is dry. Our postcondition is that we are now on the floor. So we just kind of flipped the climb ladder operator. For paint-ceiling we know that the robot has to already be on the ladder

in order to reach the ceiling so the precondition here is that the robot is on the ladder. After its painted the ceiling we have two things, the ceiling is now painted and the ladder is not dry. We haven't seen this before but this goes back to what you were saying Ashok about we have to explicitly negate things that were true in the world before. The ceiling was dry, but now it's not dry. Similarly, for painting the ladder, the robot has to be on the floor. It can't paint the ladder while it's standing on it. And the post condition is that the ladder is painted and that the ladder is not dry. This is right, David. Let us note a couple of other things. First, note how a precondition logic is providing precision to this specification or these operators. That's the power of logic, it provides clarity and precision. The other is once again note that the preconditions have only positive literals. While the postconditions may have negative literals as well. So, Ashok why is it that we can't have negative literals in our preconditions? Why can't we just say not painted ladder instead of dry ladder? David just has to do something the history of these notations. This notation comes from scripts, a planner developed in the late 60s at a Stanford research institute. One of the early robot planners that ran under ran under a robot called Shakey. Strips the planner use tier improving to form plans and it turned out that the use of only positive requals in the preconditions made the cure improving processes more efficient. This conventions is just stayed in AI since the times of strips and Shakey.

12 - Planning and State Spaces
[Click here to watch the video](#)

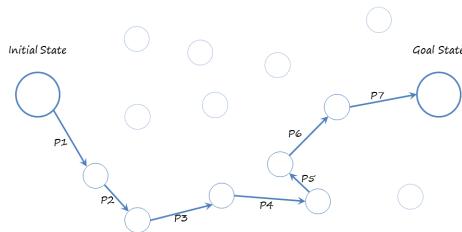


Figure 487: Planning and State Spaces

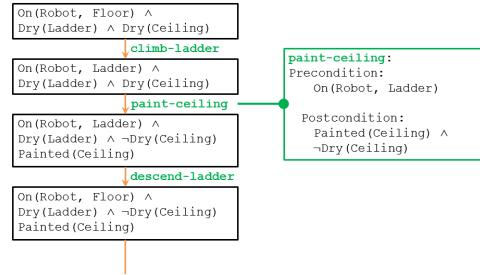


Figure 488: Planning and State Spaces

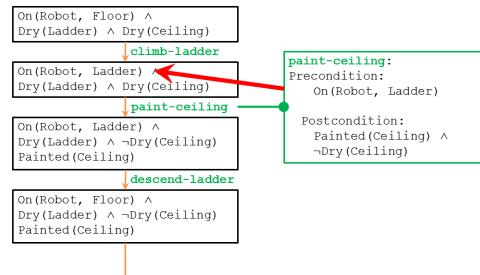


Figure 489: Planning and State Spaces

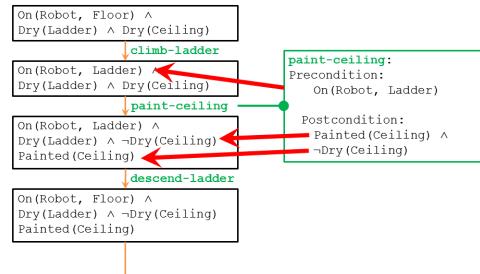


Figure 490: Planning and State Spaces

in. That's good David, and to builder's example, what do we actually do, when we have to? Plan a navigation route to go from one location, to another location, in an urban area. We use knowledge of the goal. The goal tells us, what turn to take at every intersection? We want to take a turn, that helps us get closer to the goal. So one thing we are learning here, is there are different kinds of knowledge. There is knowledge about the world, the intersections and the turns, the states and the operators more generally. There's also asset knowledge about how to do the operator selection, how to select between greatest terms at any intersection? This

knowledge is tacit, and is sometimes called control knowledge. Goals provide us with the control knowledge, of deciding how to select between different operators. Let us recall how means-end analysis work. How goals poetic control knowledge, and means-end analysis heuristic method, if you would compare the Current State and the Goal State and enumerate the differences between them. Then we'll select the operator that will help us reduce, the largest difference within the Current State and the Goal State. That's one, way of using goals as control knowledge to select between operators. Planning, provides more system mathematics matters for selecting between different operators. So the real problem now becomes, how to do operator selection, which is the same problem as, how to do action selection. Recall with me, we're talking about intelligent agents. We define intelligent agents, as agents that map perceptual history into actions. Action selection was a key problem, was a central problem. This is where planning is central, because it deals starkly with action selection, or with operator selection. Operators simply mental representations of actions, that we live with in the world. So, let us look at what a plan might look like, in the language we have been developing for planning. A plan might look like this, here is the Initial State, and a set of successor states. A series of states, punctuated by an operators that transform one state into another. Here we have expanded this operative, paint ceiling, to specify its peak conditions and post conditions, and there's several things not worthy here. Note that the preconditions of this operator, exactly match the predecessor's state. So, we have on robot ladder, here, and we have on robot ladder, here. So, some assertions of the world are true, here. And those assertions match the precondition, which is why this operator is applicable. Similarly, the post conditions of this operator, directly match the assertions about the world in this successor state. So I have painted ceiling here, there is painted ceiling there. There is not dry ceiling here, there is not dry ceiling here. So this provides a very precise way, of specifying the states, and the operators, and the exact connections between them.

13 - Planning

[Click here to watch the video](#)

Goal State:
 $\text{Painted}(\text{Ladder}) \wedge \text{Painted}(\text{Ceiling})$

$\boxed{\text{On}(\text{Robot}, \text{Floor}) \wedge \text{Dry}(\text{Ladder}) \wedge \text{Dry}(\text{Ceiling})}$

$\downarrow \text{paint-ladder}$

$\boxed{\text{On}(\text{Robot}, \text{Floor}) \wedge \neg \text{Dry}(\text{Ladder}) \wedge \text{Dry}(\text{Ceiling}) \wedge \text{Painted}(\text{Ladder})}$

Figure 491: Planning

Goal State:
 $\text{Painted}(\text{Ladder}) \wedge \text{Painted}(\text{Ceiling})$

$\boxed{\text{On}(\text{Robot}, \text{Floor}) \wedge \text{Dry}(\text{Ladder}) \wedge \text{Dry}(\text{Ceiling})}$

$\downarrow \text{paint-ladder}$

$\boxed{\text{On}(\text{Robot}, \text{Floor}) \wedge \neg \text{Dry}(\text{Ladder}) \wedge \text{Dry}(\text{Ceiling}) \wedge \text{Painted}(\text{Ladder})}$

paint-ladder:
Precondition:
 $\text{On}(\text{Robot}, \text{Floor})$

Postcondition:
 $\text{Painted}(\text{Ladder}) \wedge \neg \text{Dry}(\text{Ladder})$

Figure 492: Planning

Goal State:
 $\text{Painted}(\text{Ladder}) \wedge \text{Painted}(\text{Ceiling})$

$\boxed{\text{On}(\text{Robot}, \text{Floor}) \wedge \text{Dry}(\text{Ladder}) \wedge \text{Dry}(\text{Ceiling})}$

$\downarrow \text{paint-ladder}$

$\boxed{\text{On}(\text{Robot}, \text{Floor}) \wedge \neg \text{Dry}(\text{Ladder}) \wedge \text{Dry}(\text{Ceiling}) \wedge \text{Painted}(\text{Ladder})}$

paint-ladder:
Precondition:
 $\text{On}(\text{Robot}, \text{Floor})$

Postcondition:
 $\text{Painted}(\text{Ladder}) \wedge \neg \text{Dry}(\text{Ladder})$

climb-ladder:
Precondition:
 $\text{On}(\text{Robot}, \text{Floor}) \wedge \text{Dry}(\text{Ladder})$

Postcondition:
 $\text{On}(\text{Robot}, \text{Ladder})$

Figure 493: Planning

Let us return to means-ends analysis for just another minute. Just to see how means-ends analysis, might try to work with this problem, and get into difficulties. So this is the goal state, painted ladder and painted ceiling. And this is the initial state. Now means-ends analysis may enumerate the operators that have to deal with the painted ladder and the painted ceiling. Here the operator might be paint ladder. Here the operator might be paint ceiling, but that requires some pre-condition climb up ladder which is not

satisfy the initial state. So maintenance analysis picked the goal painted ladder. And select the operator paint ladder, which gets the maintenance owner to this state. This is the holistic method. Here is the paint-ladder specified at the right, and you can see the peak conditions of paint-ladder, match the initial state, and the post conditions match the successive state. Now that means since analysis has achieved the first goal of painted ladder it make turn to the second goal of painted ceiling. Recall that this is the current state. So, mean sense analysis may pick the operator climb-ladder, as a peak condition for the operator of painted ceiling. But note what happens, when precondition of climb ladder, constitutes a postcondition of paint ladder. So this is not dry ladder, this is quest dry ladder. There is a conflict here. In a situation like this now, the robot, would need to just wait for the ladder to become dry again, before climb ladder is a [How may an intelligent actually]. So it seems as, as if the people who are sometimes hired for working on a home or using main sense analysis. The first being the ladder, then the goal weight, until the ladder dries up, and then they of course expect me to pay them for their time. To summarize, we have a plan for achieving one of the goals, Painted Ladder. But this particular plan clobbers achieving the other goal, Painted(Ceiling), because it creates a condition, that makes it impossible to achieve the other goal. The question now becomes [How may an intelligent actually] reason about the conflict between these codes? How can planning systematically find out, how to organize these various operators, so that these conflicts do not occur? What we have described here, this goal clobbering, is true for all simple planners, sometimes called linear planners. Linear planner, does not try to reason about the conflict between these goals. Does not try to reason about how the plan for one goal may clobber another goal. Instead it just goes about making plans as if those goals can be achieved in any order.

14 - Partial Order Planning

[Click here to watch the video](#)

This introduces us to the need for parcel or-

der planning. Parcel order planning occurs with the multiple goals. And the plan for achieving one goal provost another goal. David, can you think of an example of parcel order planning? So actually I do have a kind of funny example of parcel order planning. Not long ago, we bought a couch. And to keep things simple, we decided to pay them to assemble the couch for us. Then we got the couch home and found out that once the couch was already assembled, it would not fit through the apartment door. [LAUGH] So we had to completely disassemble the couch, move the pieces in one by one, and then reassemble the couch. So the plan for assembling the couch clobbered the goal of getting the couch into the apartment. Good example. Next we will discuss how partial order planning can help us detect conflicts like this and avoid them.

15 - Partial Planning

[Click here to watch the video](#)

Goal: Painted(Ladder)



Figure 494: Partial Planning

Now let us see how partial order planning, sometimes also called nonlinear planning, may work for our ladder and ceiling problem. So here is a goal state, painted ladder. There is the initial state. We can now use the goal knowledge as control knowledge to select between different operators available in this world. The only operator whose post conditions match the goal condition of painted ladder. And whose preconditions are compatible with the initial status, paint-ladder. So we'll select that operator. When we think of applying the operator paint-ladder to the initial state, we get this as a successor state. Painted ladder and not dry ladder are coming from the post conditions of paint-ladder. Robot and floor,

and ceiling, dry, have been propagated from the initial state. We changed dry ladder to not dry ladder because that was the post condition of paint-ladder. We did not change the on robot floor and dry ceiling because pain ladder was silent about them.

16 - Exercise Partial Planning

[Click here to watch the video](#)

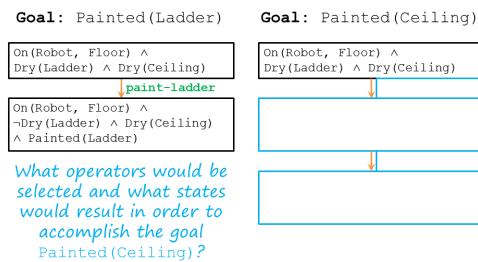


Figure 495: Exercise Partial Planning

Now that we have seen how a simple planner may work for this goal, let us see how the simple planner, the linear planner may work with the goal of painted ceiling. Please write down the operators in these boxes and the states that will be achieved after the application of these boxes in these bigger boxes.

17 - Exercise Partial Planning

[Click here to watch the video](#)

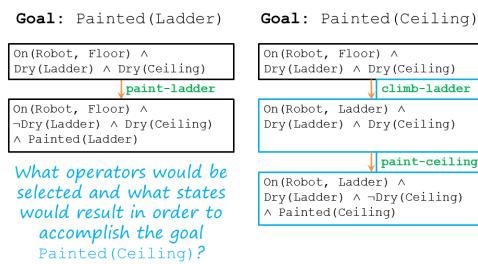


Figure 496: Exercise Partial Planning

What did you write, David? So I can tell that the first thing the robot is going to have to do to paint this ceiling is climb the ladder. So I'm first going to apply the climb ladder operator, that's going to result in the robot being

on the ladder, but the state of the ladder and the ceiling haven't changed. And then now that the robot is on the ladder, it'll apply the paint ceiling operator, which results in painted ceiling added to the state and the dry ceiling negated. That's good, David. So this is the structure of the final plan. Another question to ask here is how might a robot come up with such a plan? So let's do the same thing that we have done here earlier. We're going to use goal as a control knowledge to select an operator. So we know that we want to have the ceiling painted. So the question then becomes of the available operators, which operator has a post condition painted ceiling. And the only operator who's post-condition ceiling is the operator, paint-ceiling. So we set up that operator. So we're working backwards here. Painted-ceiling was a good condition, we selected paint-ceiling as the operator. But this paint-ceiling now has some pre conditions. One of the pre conditions is On Robot Ladder, which is different from On Robot Floor in the initial state. So now the question becomes, how can we achieve this sub goal of Robot on Ladder? If we use that sub goal as the control knowledge, then we want to select an operator whose post conditions will match this top goal and that operator is climb ladder. So now we have that climb ladder here and the preconditions of climb ladder match this initial state and that is how we get to this plan, climb ladder and paint ceiling. So we were working backwards using goal knowledge as control knowledge to select between operators. So note that we just made a connection back to problem reduction that we talked about right after means and analysis. Ashok in his description talked about the sub goal of getting up the ladder. When we talked about problem reduction earlier, we talked about the need to break big problems down into smaller problems. But we didn't exactly talk about how an agent would go about doing that. Here we see one way in which an agent would go about actually identifying those subgoals in order to accomplish the final goal.

18 - Detecting Conflicts

[Click here to watch the video](#)

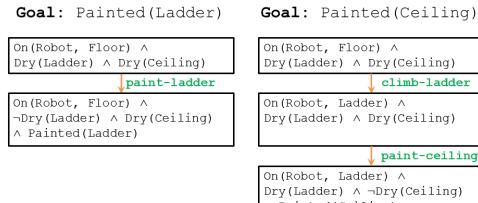


Figure 497: Detecting Conflicts

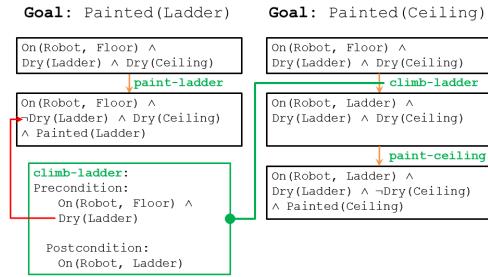


Figure 501: Detecting Conflicts

Detecting Conflicts

For each precondition in current plan:

- If precondition for an operator in the current plan is clobbered by a state in another plan:
- Promote current plan above other plan

Figure 498: Detecting Conflicts



Figure 502: Detecting Conflicts

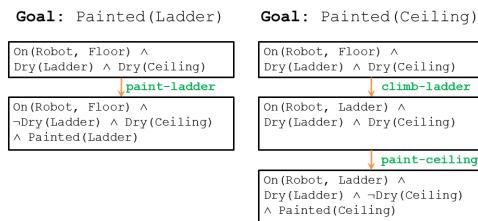


Figure 499: Detecting Conflicts

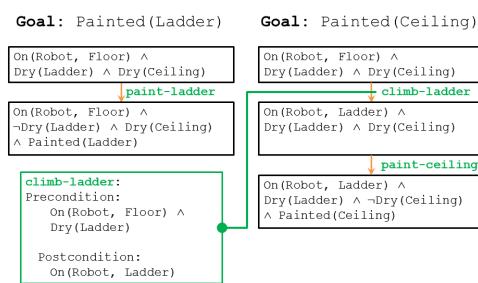


Figure 500: Detecting Conflicts

So what the partial or the planner has done so far is to view the two goals as if they were independent of each other. And come up with a partial plan for each of the two goals. It has not yet detected any conflicts between that will not resolve those conflicts. The next thing would be to examine the relationship between these two plans and see if there are any conflicts between. But how might a partial order plan go about detecting conflicts between two plans? So, here is plan one imagined, here is plan two. The partial order planner may go about detecting conflicts. We're look at each precondition of the current plan. Under the precondition of an operator any current plan is clobbered by some state in the, another plan, in the second plan, than the partial order planner would know that there's a conflict between them. [Then it can go about] goal resolving these conflicts, but promoting or demoting one clients goal or another clients goal. There's if some stated plan B covers the application of some operator in plan A, then we now want to alter the goals in this plan and this plan in such a way that this operator's done before that state is achieved. Now, let us see how the partial order planner may go about detecting

conflicts within these two plans. So the partial order planner may begin with this plan for painting the ladder. And see whether the precondition of this operator, paint-ladder, are clobbered by any state in the second plan. As it turns out, that doesn't happen in this example. Now, the partial order planner will look at the operands in the second plan. And see whether the preconditions of any of the operators are clobbered by some status in this first plan. So let's look at climb-ladder here. The precondition of climb-ladder is, on robot, floor, and dry ladder. And as this precondition is compared with the states. In the first plan, we can eventually see the conflict. Here is dry ladder, and here is not dry ladder. And this way the partial order planner has been able to find that the water-less states here in the first plan proverbs the precondition of one operator on this second plan. To resolve this conflict, the partial order planner may promote these goals or the goal of painting the ladder. Some of you also noticed that after the robot has painted the ceiling, the robot is on ladder. But in order to apply the paint ladder operator, the robot must be on the floor. So here there is an open condition problem. This particular condition where this operator is not being satisfied. When the robot is on the ladder. We'll come to this in a minute.

19 - Open Preconditions

[Click here to watch the video](#)

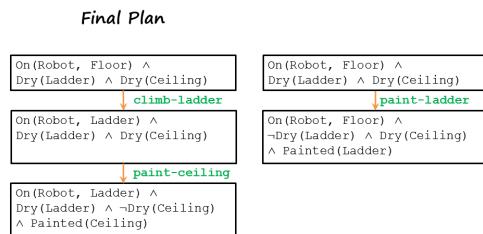


Figure 503: Open Preconditions

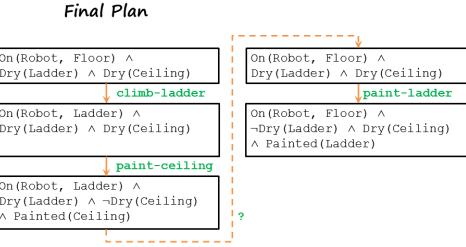


Figure 504: Open Preconditions

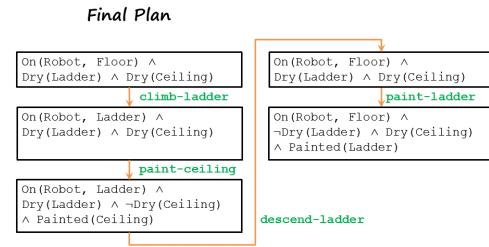


Figure 505: Open Preconditions

So recall that in order to resolve the conflict, the partial order planner has decided to promote this goal over that one. As it tries to connect these two plans, it finds that there is an open condition problem that we just talked about On Robot Ladder, does not match On Robot Floor. So now it needs to select an operator whose first condition will match this state. Robot On Floor. And those three conditions will match this state, Robot On Ladder. And there is just one operator that matches those conditions. And that operator is descend ladder. So now the partial order planner uses this information to select the operator, the simulator, and now we have a complete plan. So now you know about the algorithm for partial order planning, and how it works in practice. But what does this tell us about intelligence? Let's consider several postulates. First, knowledge is not just about the world. Knowledge is also controlled knowledge. It is often tacit, but this controlled knowledge helps us select between operators. Second, that goals provide control knowledge. Goals can be used to decide between different operators, and we select an operator that helps us move closer to the goal. Third, we can view partial order planning

as an interaction between several different kinds of agents or abilities. Each agent here represents a small micro ability. There is this agent which was responsible for generating plans for each of the goals independently, then there was an agent responsible for detecting conflicts between them. Then there was a third agent responsible for resolving this conflict. So we can think of partial order planning as emerging out of interaction between three different agents, each one of which is capable of only a small ability. So we can think of partial order planning as emerging out of interaction between free agents, where each agent is capable of only one small task. Minsky has proposed a notion of a society of mind. A society of agents inside an intelligent agent's mind that work together to produce complex behavior, where each agent, itself is very simple. As in this case, a simple agent for detecting conflict, or a simple agent for resolving conflicts, and of course an agent for making simple plans with simple goals. It is one other lesson to take away from here. When you and I solve problems like the ladder and the ceiling problem, we seem to address these problems almost effortlessly and almost instantaneously. So it looks really simple. What AI does, however, is to make the process explicit. To write a computer program that can solve the same problem is very hard. It is hard because the computer program must specify each operator, each precondition, each state, each goal, every step very, very clearly and very, very precisely. By writing this computer program is this AI agents that consults problem. We make the process that humans might be using more explicit. We generate hypotheses about how humans might be doing it, which is a very powerful idea.

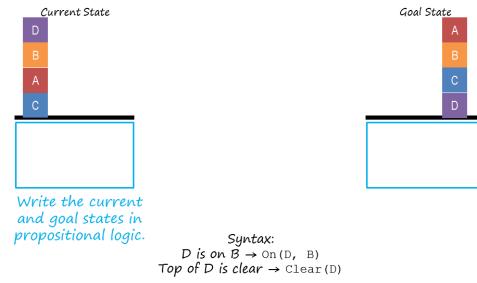


Figure 506: Exercise Partial Order Planning I

Now that we have seen partial art of planning in action, let us try to do a series of exercises to make sure that we understand it clearly. We have come across this problem earlier. This is the micro world of blocks. Here is the initial state. And here is the goal state. We need to transfer from this initial state to the goal state, moving only one block at a time. Please write on the initial state and the goal state in propositional logic.

21 - Exercise Partial Order Planning I

[Click here to watch the video](#)

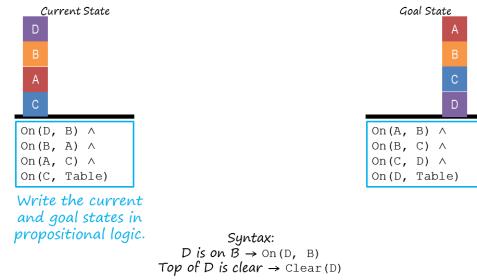


Figure 507: Exercise Partial Order Planning I

David? So our initial state, is that each block is on top of the other. D is on B. B is on A. A is on C. And C is on the table. Our goal state, is each block on top of the other in a different order, an alphabetic order. So, A is on B. B is on C. And C is on the table. So our initial state is that the blocks are all stacked up, D is on B, B is on A, A is on C. C is on the table. And our goal state, is that the blocks are stacked up in alphabetical order, so A is on B, B is on C, C is on D and D is on table.

20 - Exercise Partial Order Planning I

[Click here to watch the video](#)

22 - Exercise Partial Order Planning II

[Click here to watch the video](#)

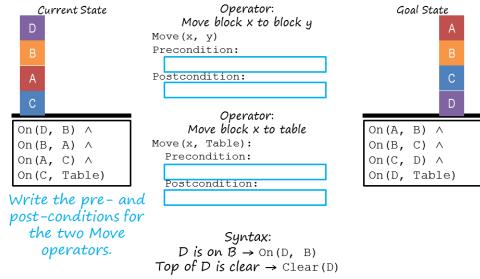


Figure 508: Exercise Partial Order Planning II

Now that we humans find addressing problems like this almost trivial, we know what to do here. Put D on the table, put B on the table, and so on. And then put C on top of D and so on. The question is, how can we write an AI program that can do it? And, by writing an AI program, how can we make things so precise that that will provide insight into human intelligence. To do this, let us start writing the operators that are available in this particular work. There are only two operators. I can either move block x to block y, which is the first operator here. Or I can move block x to the table. Note two things. First, instead of saying block A and block B, we have variabialized them, move block x to block y, where x could be A, B, C or D, and similarly for block y. And this is just a more concised notation. Second, that in order to move block x to block y, both x and y must be clear. That is neither x nor y should have any of the block on top of them. Given this setup, please write down the specification of the first operator as well as the second operator.

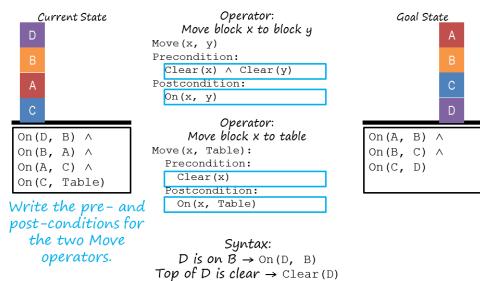
23 - Exercise Partial Order Planning II[Click here to watch the video](#)

Figure 509: Exercise Partial Order Planning II

So like you said, our precondition for the first one, is that both x and y are clear. We can't move x if there's anything on top of x, and we can't put it on y if something is already on top of y. Our post condition then, is that x is on y. For the table it's a little bit easier, the table has unlimited room. So for the table, as long as X is clear we can move X to the table. And in the postcondition is that X is now on the table.

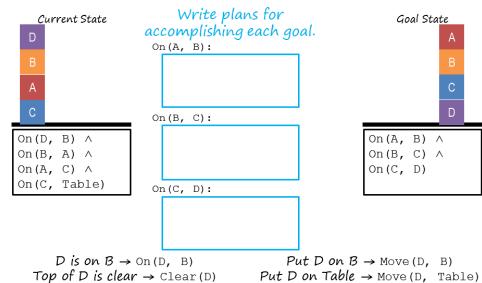
24 - Exercise Partial Order Planning III[Click here to watch the video](#)

Figure 510: Exercise Partial Order Planning III

So given the various goals here, A and B, B and C, and so forth, write down the plan for accomplishing each goal, as if these goals were independent of each other. We are shown here only three goals here not the fourth goal of D on table, because of lack of space. But D on table anyway is [trivial].

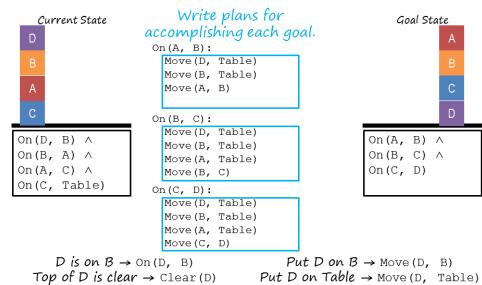
25 - Exercise Partial Order Planning III[Click here to watch the video](#)

Figure 511: Exercise Partial Order Planning III

So like you said [Ashok], the plan of putting D on table's kind of trivial. And we actually see that it's the first step of any other plan. So we don't really need to articulate that explicitly.

For putting A on B, our fastest idea would be to put D on the table, then to put B on the table, then to put A on top of B. I would just get a straight to putting A on B. For putting B on C, we need to put D on the table, B on the table, A on the table, and then move B on to the top of C. And then, for putting for C on D, we would need to move D to the table, B to the table, A to the table, and then put C on top of D.

26 - Exercise Partial Order Planning IV

[Click here to watch the video](#)

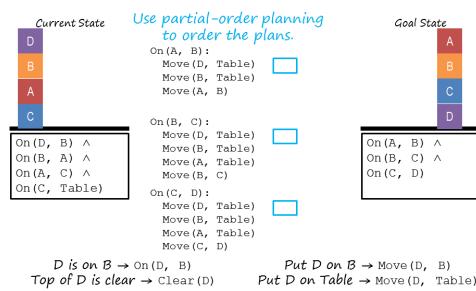


Figure 512: Exercise Partial Order Planning IV

Now that we have these three plans for accomplishing the three goals, can you detect the conflicts between these plans? Use a pencil and a piece of paper, to detect the conflicts and resolve the conflicts and then write down the ordering of the goals in these boxes.

27 - Exercise Partial Order Planning IV

[Click here to watch the video](#)

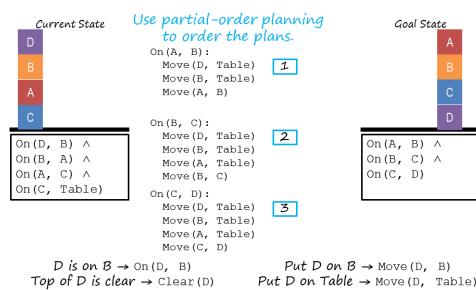


Figure 513: Exercise Partial Order Planning IV

What ordering did you come up with David? So what we see is that each goal actually clobbers the next goal in its last step. So the post

condition of move A, B is that A is now on B. But a precondition for move B, C is that B is clear. Because A is on B, B is no longer clear, so we can't move B to C, so this plan has clobbered this plan. Similarly, if we move B to C, we no longer can move C on top of D because C is no longer clear, so this plan has clobbered this plan. So we're going to need to do this plan first, then this plan, then this plan. Good, David.

28 - Exercise Partial Order Planning V

[Click here to watch the video](#)

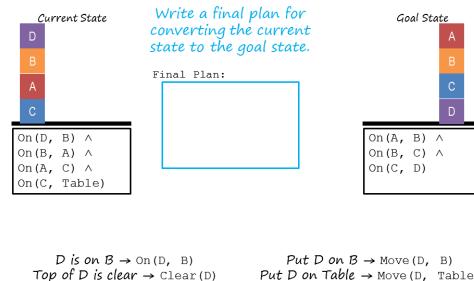


Figure 514: Exercise Partial Order Planning V

Now that we know about the conflict between these plans, please write down the final plan for achieving the goal state. To save space, just write down the operators. You don't have to specify all the states in this plan.

29 - Exercise Partial Order Planning V

[Click here to watch the video](#)

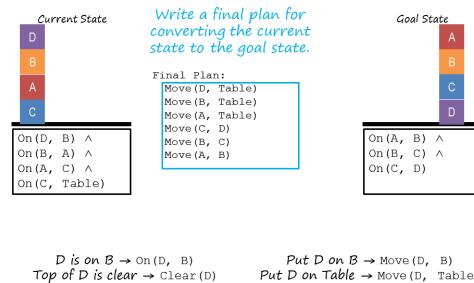


Figure 515: Exercise Partial Order Planning V

That's good David. Note that when we did this problem previously using means analysis and product reduction, we ran in to all kinds of problems, because plans for property goals and we had no way of ordering these radius goals. Now we have a way, that's the power of partial

[order planning]. Note also that this partial art of planning, this algorithm makes certain things that are implicit in human reasoning, explicit. Presumably when you and I reason about things we must be reasoning something like this, or at least this is one hypothesis about how we might be reasoning. And we have just made many of the operations of human reasoning so explicit and so precise.

30 - Hierarchical Task Network Planning

[Click here to watch the video](#)

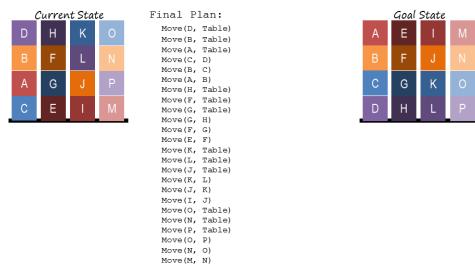
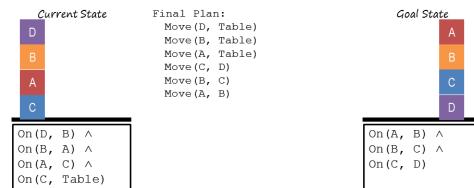


Figure 516: Hierarchical Task Network Planning

Our next topic in planning is called hierarchical planning. We'll introduce the idea to you. We'll also introduce the representation called hierarchical task network to you. HTN for hierarchical task network. To illustrate hierarchical planning, imagine that you are still in the box microworld. Here is the initial state. And here is the goal state. These states are more complicated than any initial state and goal state that we have encountered so far. So as previously, we can use partial order planning to come up with a plan to go from this initial state to goal state. Here is the final plan, and as you can see, it's pretty long and complicated, with a large number of operations in them. So the question then becomes, can we abstract some of these operations at a higher level? So that instead of thinking in terms of these slow level move operations, we can think in terms of high level macro operations. And those macro operations will then make the problems space much smaller, much simpler so that we can navigate it. And then we can expand those macro operators into the move operations.

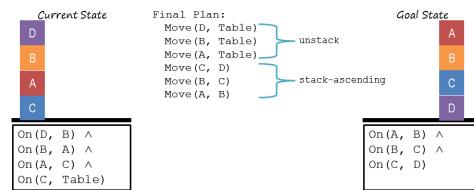
31 - Hierarchical Decomposition

[Click here to watch the video](#)



Put D on B → Move(D, B)
Put D on Table → Move(D, Table)

Figure 517: Hierarchical Decomposition



Put D on B → Move(D, B)
Put D on Table → Move(D, Table)

Figure 518: Hierarchical Decomposition

unstack:
Precondition:
On(w, x) ∧
On(x, y) ∧
On(y, z) ∧
On(z, Table)
Postcondition:
On(w, Table) ∧
On(x, Table) ∧
On(y, Table) ∧
On(z, Table)
Method:
Move(w, Table)
Move(x, Table)
Move(y, Table)

stack-ascending:
Precondition:
On(a, Table) ∧
On(b, Table) ∧
On(c, Table) ∧
On(d, Table)
Postcondition:
On(a, b) ∧
On(b, c) ∧
On(c, d) ∧
On(d, Table)
Method:
Move(c, d)
Move(b, c)
Move(a, b)

Figure 519: Hierarchical Decomposition

So look at the macro operators at a high level abstraction, consider this one part of the current problem. Here is the initial state, here is the goal state, there is the final plan. To enlist this idea of macro operators, and hierarchical planning, at multiple [levels] of abstraction, let us read with this problem that we had encountered earlier. This was the initial state, this was the goal state. And we come up, we came up with this as the final plan. Now, we can think of these three operations as being abstracted out

into a macro operator that we can call unstack. And these three operations being abstracted out into a macro operator that we can call stack-ascending. Just simply saying stacking them in a particular ascending order. Here is the specification of the two macro operators. Unstack, and stack-ascending. You do preconditions and post conditions. And this macro operator, also tells you how this macro operator can be expanded in to the lower level move operations. Similarly for the stack ascending macro operator.

32 - Hierarchical Planning

[Click here to watch the video](#)

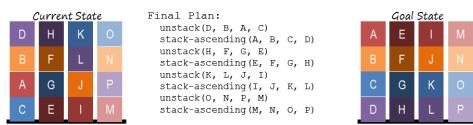


Figure 520: Hierarchical Planning

Now that we have illustrated hierarchical planning, what does it tell us about intelligence? Intelligent agents, both cognitive and artificial, constantly faced with large, complex problems. The problem spaces corresponding to these problems often have [combinatorial] explosion of states in them. Intelligent agents address these complex problems by thinking at multiple levels of abstraction. So that at any one level of abstraction, the problem appears small and simple. In order to be able to reason at these multiple levels of abstraction, we need knowledge at multiple levels of abstraction. In this case, there was knowledge not only at the level of move operations, but also the level of macro operations, like unstack and stack ascending. And perhaps even higher level macro operations, like sort. This goes back to the fundamental notion of knowledge based AI. Intelligent agents use knowledge in order to be able to tackle hard, complex problems.

33 - Assignment Planning

[Click here to watch the video](#)

Assignment

How would you use planning to address Raven's progressive matrices?

Figure 521: Assignment Planning

How would you use planning to develop an agent that can answer Raven's progressive matrices? So, the first question you want to ask here is what are our states? What's our initial state? And what's our final state? Given that, what are the operators that allow the transition between them. How would we select those operators? We are talking about partial ordering planning in this lesson, what conflicts are possible when we are trying to solve Raven's problems? How would we detect those conflicts beforehand and avoid them? Note that again we can consider this at two different levels. First, we can think of the agent as having a plan for how to address any new problem that comes in. Or second, we can consider the agent as discerning the underlying plan behind a new problem.

34 - Wrap Up

[Click here to watch the video](#)

To recap...

- Planning in propositional logic
- Goal conflicts
- Partial-order planning for conflict avoidance
- Hierarchical task networks

Figure 522: Wrap Up

So today we've discussed how to plan out actions using formal logic. We started off by talking about states, operators, and goals in formal logic. We then used those to contextualize our discussion on detecting conflicts that might arise.

This introduced the need for partial-order planning which helps us avoid those conflicts beforehand. Finally we talked about hierarchical tasks networks which can be used for hierarchical planning. Now, we're going to move on to understanding, which builds on our notion of frames from a few lessons ago, but if you're interested in this, you can jump forward to our lessons on design. Configuration and diagnosis leverage some of the concepts of planning very heavily.

35 - The Cognitive Connection

[Click here to watch the video](#)

Planning is another process central to cognition. It is central because action selection is central to cognition. You and I are constantly faced with the problem of selecting actions. Where should I go for dinner today? What should I cook for dinner today? How do I cook what I wanted to cook? I got a bonus. What should I do with the bonus? Shall I go for a vacation? How should I go to a vacation? Where should I go to a vacation? These are all problems of action selection. And I need planning to select the appropriate actions. Cognitive agents also have multiple goals. As a professor, one of my goals right now is to talk with you. Another goal that I have is to become rich, although I know that becoming a professor is not going to make me rich. The point is that cognitive agents have multiple goals that can have interactions between them. Sometimes interaction is positive. Achieving one goal provides an opportunity for achieving the second goal. Sometime the interaction is negative, there are conflicts. Cognitive agents detect those conflicts. They avoid those conflicts. And planning, then, is a central process for achieving multiple goals at the same time.

36 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

37 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

Planning uses states, operators and goals in formal logic. Partial-order planning helps us avoid conflicts in advance. Hierarchical Task Networks are used for Hierarchical Planing. Configuration and Diagnosis leverage various concepts of planning.

References

1. Winston P., Artificial Intelligence, Pages 323-338.
2. Russell, S., & Norvig, P. Artificial Intelligence: A Modern Approach, Section 11.3.

Optional Reading:

1. Winston Chapter 15, pages 323-336; [Click here](#)
2. Russell & Norvig Chapter 11, Section 3; Russell Norvig Section 11.3.pdf

Exercises

Exercise 1:

The original STRIPS program for planning was designed to control Shakey, one of the first AI robots to combine action, perception and cognition. Shakey's world consisted of rooms (say, two rooms) lined up along a corridor, where each room had only one (always open) door and a light switch. By convention, a door between a room and the corridor was considered as being in both of them. Although Shakey was not dexterous enough to climb on a box or toggle a switch, the STRIPS planner was capable of forming plans for climbing on a box and switching lights on/off. Shakey's six actions were the following:

Go (x,y) which requires that Shakey be at x; x and y are locations in the same room.

Push (b, x, y): Push box b from location x to location y.

ClimbUp(b) and ClimbDown(b): climb onto a box and climb down from a box, respectively.

Lesson 14 - Understanding



In mathematics you don't understand things. You just get used to them.
– John von Neumann.

01 - Preview

[Click here to watch the video](#)

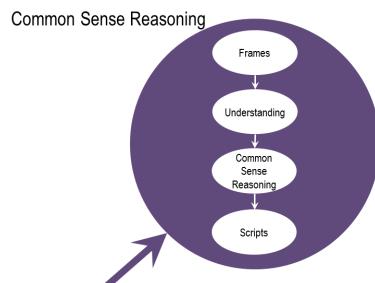


Figure 523: Preview

lesson. We'll start with talking about thematic role systems, which are the most structured type of frame representation. Then we'll talk about how these roles, these thematic roles can be used to resolve ambiguity in understanding the world. Finally, we'll talk about how grammar and other constraints can be used to guide our interpretation of the world.

02 - The Earthquake Report

[Click here to watch the video](#)

Lesson Preview

- Thematic role systems
- Ambiguity
- Constraints

Story 1:
Today, an extremely serious earthquake of magnitude 8.5 hit Lower Slabovia, killing 25 people and causing \$500 million in damage. The President of Lower Slabovia said that the hard-hit area near the Sadie Hawkins fault has been a danger zone for years.

Story 2:
Today, the President of Lower Slabovia killed 25 proposals totaling \$500 million for research in earthquake prediction. Our Lower Slabovian correspondent calculates that 8.5 research proposals are rejected for every one approved. There are rumors that the President's science advisor, Sadie Hawkins, is at fault.

Figure 525: The Earthquake Report

Figure 524: Preview

Today, we will talk about understanding. Understanding is a very big general world. We'll be talking about understanding of even some stories in the world. Understanding will be very heavily on what we learn about Frames. And it will help us set up the infrastructure for learning about common sense reasoning in the next

Sentence 1:
A serious earthquake killed 25 people in Lower Slabovia.

Sentence 2:
The President of Lower Slabovia killed 25 proposals for earthquake prediction.

Figure 526: The Earthquake Report

We'll use story understanding, To examine the general processes of understanding. Here are two stories that we have encountered earlier. The first story is talking about the earthquake that hit lower Slabovia and caused some damage. The second story is talking about the President of lower Slabovia who killed a number of proposals. Both of them deal with killing. But the meaning of the two stories are very, very different. Humans make use of stories to understand the world around them. The world offers a very large amount of data at any given time. We use the stories to provide structure to that data, to make coherent sense of it. When we discuss these two stories earlier, using frame as a knowledge representation language, we found that we could have a frame of this notion of killing, killing of 25 people, and this notion of killing, killing of 25 proposals. And there was no simple way in the frame knowledge representation to disambiguate between those two meanings of killing. Now we'll examine how we can construct a different interpretation of the first story, or the interpretation of the second story, based on different interpretations of the word killing in the two stories. In order to make the two stories simpler, and to illustrate our point, we're going to focus on one sentence from each story, sentences that contain the word kill. Humans that have lived through difficulty and understanding the first sentence means completely different from the second sentence because the killed in the first sentence means something completely different from the killed in the second sentence. How can we build an err program that can do the same thing? We will see that err program will need to use a lot of background knowledge in order to be able to do that, and that would generate hypotheses about how humans might be using that background knowledge to similarly disambiguate between different senses of the word killed.

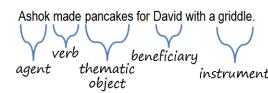
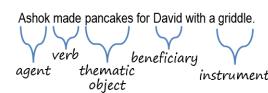


Figure 527: Thematic Role Systems



Thematic Role
verb : make
agent : Ashok
beneficiary : David
thematic object : pancakes
instrument : griddle

Figure 528: Thematic Role Systems

Let us first consider a simpler sentence. So consider the sentence Ashok made pancakes for David with a griddle. I'm sure you understood the meaning of this sentence almost immediately. But what did you understand? What is the meaning of meaning? We can do several different kinds of analysis on this sentence. We can do lexical analysis, which will categorize each of these words into different lexical categories. For example, Ashok is a noun, made is a verb, pancakes is a noun, and so on. We can do syntactic analysis. In terms of the structure of this particular sentence. So, you might say Ashok is a non phrase. Made pancakes for David with a griddle is a verb phrase. And this particular verb phrase itself has sub phrases in it. Or we can do semantic analysis on this, and say that Ashok was the agent. Made was the action. Pancakes were the object that got made. David was the beneficiary. Griddle was the instrument. The knowledge base here as oppose to understanding stories like this. Semantic analysis is at a forefront. Syntactic analysis and lexicon analysis will serve semantic analysis. So here are some of the semantic categories in terms of which we can classify the different words in the sentence.

03 - Thematic Role Systems
[Click here to watch the video](#)

So Ashok is an agent, made is an action or verb in the lexical sense, pancakes are the thematic objects, the things that are getting made and so on. The frame for representing understanding of this sentence has the verb of the action make, the agent Ashok, and so on just like we just discussed. This then is the meaning of meaning. This is what the agent understands when it understands the meaning of this sentence. This is perhaps also what you understand when you understand the meaning of this sentence. How do we know that you understood the meaning of this sentence? Well, we know that because I can ask you some questions and you can draw the right kind of inferences from it. So for example, given this sentence, I can ask you, well who ate the pancakes? And you might be able to say, well David ate the pancakes because Ashok made the pancakes for David. Notice that this information about who ate the pancakes was not present in this particular sentence. This is an inference you're drawing. This is a very similar to what we had encountered earlier when we had sentence like Ashok ate a frog. At that time too, we'd ask questions like, well, was Ashok happy at the end? And the frame had some default values which said Ashok was probably happy. Or, was the frog dead at the end, and the frame for eating had some default value which said that the frog was dead at the end? So according to this theory, the meaning lies in the inferences we can draw from it. You understand the meaning of this if you can draw the right inferences. You do not understand the meaning of this if you cannot draw the right inferences or if you can draw only the wrong inferences. This frame representation of the meaning of this particular sentence allows you to draw the right inferences. Given the action make here, the thematic role pertains to the relationship of various words in the sentence. To this particular action of making. Ashok is the agent, David is the beneficiary, and so on. So far we are describing meaning of this sentence, and how we can capture that meaning in this frame. We have not yet described the process by which this knowledge is extracted out of that sentence. The extraction of the meaning of this sentence is exactly the topic that we will discuss next.

04 - Thematic Role Systems

[Click here to watch the video](#)

In order to start resolving ambiguities of the kind the occur with the earthquake story, we need a new knowledge representation called a thematic rule system. A thematic rule system is a type of frame system, where the frame represents an action or event identified by a word. The word caused forth a number of expectations about the roles that are connected with that particular event. David, can you think of an example of the thematic-role frame? So one simple example that comes to mind for me is the word throw. I know that if a throw action has taken place, a few things have to be in play. I expect there to be someone doing the throwing. I expect something to be thrown. I might expect a target specified by the word at or a destination specified by the word to. So, for example, I throw the ball to a choke, or I throw the ball at a choke, depending on how nice I'm feeling. [LAUGH] Note that even though David isn't talking about a specific instance of throwing, he's still able to generate expectations with the general action of throwing. This shows what a thematic case frame does for you. It is able to generate expectations. Let us look at how it would actually work in action.

05 - Exercise Thematic Role Systems

[Click here to watch the video](#)

David went to the meeting with Ashok by car.

Thematic Role	
verb :	<input type="text"/>
agent :	<input type="text"/>
coagent :	<input type="text"/>
destination :	<input type="text"/>
conveyance :	<input type="text"/>

Write the thematic role frame given by the sentence above.

Figure 529: Exercise Thematic Role Systems

Now that we understand how to represent the meaning of stories, let us consider a different story. David went to the meeting with Ashok by car. Please write down the meaning of this story, in terms of the slots of this particular thematic role frame.

06 - Exercise Thematic Role Systems

[Click here to watch the video](#)

David went to the meeting with Ashok by car.
Thematic Role
verb : go
agent : David
coagent : Ashok
destination : meeting
conveyance : car

Write the thematic role frame given by the sentence above.

Figure 530: Exercise Thematic Role Systems

What did you write, David? So we start with a verb, and the verb here is went, which is the past tense of the action go. Our agent here is David, myself, because I'm the one doing the action, I'm the one going. The coagent is Ashok, because I'm going with Ashok. The destination is to the meeting, and the conveyance here is by car, so car is the conveyance. That's right, David. But how did we know that David was the agent? How did we know that the destination was the meeting? How did we know that car was the conveyance? That's what we'll look at next.

07 - Constraints

[Click here to watch the video](#)

David went to the meeting with Ashok by car.

Thematic Role
verb : go
agent : David
coagent : Ashok
destination : meeting
conveyance : car

Prepositional Constraints	
Preposition	Thematic Roles
by	agent, conveyance, location
for	beneficiary, duration
from	source
to	destination
with	coagent, instrument

Figure 531: Constraints

Let us use the assignment of car as the conveyance, to illustrate how these different words get assigned to different categories, different slots in this frame. Now, we know that car was a conveyance because of the role that this preposition, by, plays here. That is, an intelligent agent

might make use of the structure of the sentence to make sense of the story. We have designed human language in such a way that there is a particular structure to them. Prepositions, for instance, play a very important role. Here are some of the common prepositions: by, for, from, to, with. Each preposition plays certain thematic roles. By can be an, by an agent, or by a conveyance, or by a location. Similarly for the other prepositions. So the moment we see by, here we know that, whatever is going to come after by, the care can be a agent, or a conveyance, or a location. Note again that the categories we are using here are semantic categories, we are not saying noun, or verb or anything like that, what we are saying here is beneficiary and duration, and source which are semantic categories, that allow us to draw inferences. But how did we know that car was a conveyance, and not an agent, or a location? In general, while the structure of language does provide constraints, these constraints do not always definitely determine the meaning of that particular word. We'll use additional knowledge to find the exact meaning of the word car.

08 - Resolving Ambiguity in Prepositions

[Click here to watch the video](#)

Prepositional Constraints	
Preposition	Thematic Roles
by	agent, conveyance, location
for	beneficiary, duration
from	source
to	destination
with	coagent, instrument

Figure 532: Resolving Ambiguity in Prepositions

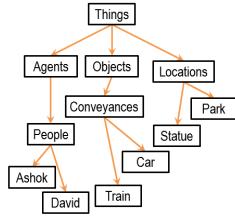


Figure 533: Resolving Ambiguity in Prepositions

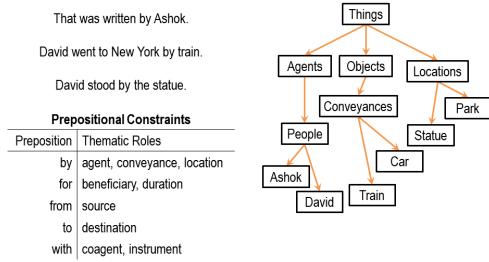


Figure 534: Resolving Ambiguity in Prepositions

But first let us look at examples of the three kinds of semantic categories that by point to. So here is a sentence in which by points to an agent. That was written be Ashok. Actually, this particular sentence was written by David. The second sentence, David went to New York by train, here by is pointing to a conveyance. David stood by the statue. Here, by it is pointing to a location. By the statue. So the use of the word by helped us, in that it constrained the interpretation of Ashok to either an agent, conveyance or location. It by itself doesn't tell us whether Ashok is an agent or a conveyance or location. We need some additional knowledge. Let us suppose that the agent has an ontology of the world. The term ontology initially comes from philosophy where it means the nature of reality. An artificial intelligence determine ontology often reference to the conceptualization of the world. The categories and terms of which I specify the world. These categories have become the vocabulary of the knowledge representation. So let us supposed that the agent has this ontology of the world. The world is composed of things, and things can be agents or objects or locations. Agents are people, and

Ashok and David are examples of agents. Objects can be conveyances. And trains and cars are examples of conveyances. Obviously this is a very small part of someone's ontology about the world. Now this ontology helps us decide that Ashok, in the first sentence, is an agent. Let's see how. Here we have by Ashok, and we know from the prepositional constraints that by can refer to an agent, conveyance, or location. So the question now becomes, is Ashok an agent, Ashok a conveyance, or Ashok a location? We can look into this ontology. Ashok is people, which is an agent. So now we know, that Ashok must be an agent. Similar the second sentence. Train can be an agent, conveyance or location. Let's look at our ontology. A train is a conveyance. So now we know the train is a conveyance. Similarly for statue. Statue in this case specifies a location. Note that this analysis applies to different prepositions, not just to by. Supposing this first sentence was, That was written with Ashok. In which case, this preposition with will point to Ashok either being a coagent or being an instrument. So again, we now know that Ashok is a coagent in the sentence which contain the preposition with. There is another important thing to note here. Initially, when we were give this sentence, David went to New York by train, we started doing bottom-up processing. David was a noun, went was a verb. But pretty soon we shifted to top-down processing. As an example, you already have this background knowledge, and this background knowledge tells us in a top-down manner that by, here, can refer to an agent, a conveyance, or a location. We also have this additional background knowledge, and this background knowledge tells us that in this particular sentence. The train describes frame to a conveyance, because train is a conveyance in this ontology. So low-level bottom up processing generates queues, which access probes into memory. Memory then returns knowledge like this, and the processing becomes top down. The top down processing tells us how to interpret the various words in this sentence, how to make sense of this story.

09 - Ambiguity in Verbs

[Click here to watch the video](#)

Of course we don't just see ambiguity in propositions, we also see it in words. David, can you think of ambiguity in a word? To go back to our throw example, I might say I was wondering why the ball was getting bigger and then it hit me. The word hit is ambiguous because it could either mean occur or strike. It occurred to me why the ball was getting bigger, or the ball physically struck me. Take another example. It's hard to explain puns to kleptomaniacs because they always take things literally. The humor here is that take could either mean interpret, as in they always interpret things literally, or it could mean physically pick up and remove. As in they literally always take things. In fact, many puns derive their humor from the fact that a word in the sentence can play two different roles in the sentence at once. And the sentence still makes sense. Did you know that I was going to say this will be much fun? The goal going forward is to look at how agents will resolve some ambiguities. When one correct meaning, and only one correct meaning is possible. Not however, that we might build our character with a bit of humor. Perhaps one form of humor is when a single sentence can be made to fill multiple grammatical simultaneously.

10 - Resolving Ambiguity in Verbs

[Click here to watch the video](#)

take. verb. /täk/
 • to reach for and hold
 • to bring or transport
 • to remove
 • to medicate
 • to capture
 • to steal
 • to cheat or swindle
 • to subtract
 • to measure
 • to accompany
 • to assume control
 • to remove from the body

Figure 535: Resolving Ambiguity in Verbs

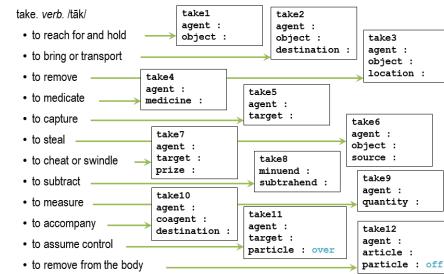


Figure 536: Resolving Ambiguity in Verbs

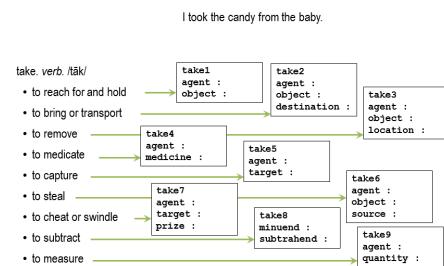


Figure 537: Resolving Ambiguity in Verbs

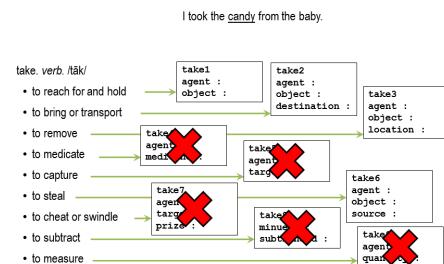


Figure 538: Resolving Ambiguity in Verbs

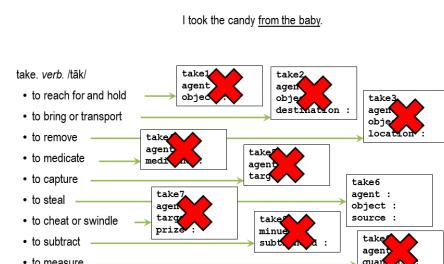


Figure 539: Resolving Ambiguity in Verbs

We saw in the previous example how sentences in a story can be ambiguous. For example, by, could have referred to an agent, a conveyance,

or a location. This is true, not just for prepositions, but is also true of other [lexical] categories, like words. In fact, words often can have several interpretations. Let us consider the word take as an example. Take is a very common word. It has at least these 12 different meanings. Consider for instance to medicate. Ashok took an aspirin. Here, the meaning is that Ashok took aspirin as a medication. Each of these meanings has a common meaning of take, as we will say in just a minute. But given a sentence in which take occurs, how do we know which of these meanings is intended by the word, take? So suppose the input sentence was, my doctor took my blood pressure. The taken in this sentence refers to, to measure and not to any of the others. Let us examine this issue further. So, for each of these 12 interpretations of take, we have a frame-like representation. So take 1 to take 12. Each of this frame-like representation specifies the thematic roles that go with that particular meaning of take. So in this particular meaning of take, take 1, we have an agent and an object. In this meaning of take, take 12. We have an agent, an article, and a particle. Another word that typically occurs with take which signifies this meaning, so to take clothes off from a body. Let us consider another example of particle. Let us consider take11. The meaning of this take is to assume control, as in to assume control of a company, or to assume control of a country. When the meaning is intended to be to assume control, then take typically occurs with the word over. Take over a company. Take over a country. So, over then is a particle that signifies this eleventh meaning of take. To go deeper into story understanding, consider the simple story I took the candy from the baby. What is the meaning of the word take here? You and I get this immediately, but how can an agent get it? To keep it simple, we have shown here just nine meanings of take, you could have added the other three as well. Although we started with bottom-up processing, we're now going to shift to top-down processing. Because there's something about the background knowledge about candy that we have, which is going to eliminate lots of choice. In particular, we know that candy is not a medicine, so

this particular choice goes away. We know that candy is not a quantity, so this choice goes away. Several of these choices disappear, because of our background knowledge of candy. Just like some of the constraints came from our background knowledge of the semantic category of candy, other constraints come from our background knowledge of the preposition, from. In the table showing prepositions earlier, from referred to a source. These three frames do not have anything to do with the source, and therefore we eliminate them. We're left only with this particular frame, which has source in it as required by the preposition from. And thus we decide that the interpretation of took in this particular sentence is to steal from a baby. And thus we infer the correct interpretation of take, in this particular sentence. It refers to, to steal. This is the only frame that is still active

11 - Exercise Resolving Ambiguity in Verbs

[Click here to watch the video](#)

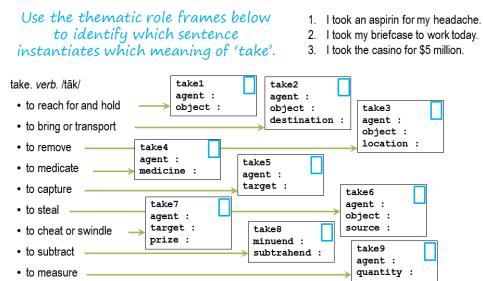


Figure 540: Exercise Resolving Ambiguity in Verbs

Now that we have examined how the thematic role frames help us disseminate between different meanings of take, let us do some exercises together. In fact, let's do three exercises together. Here are three sentences. Please pick a box which best captures the meaning of take in each of these three sentences.

12 - Exercise Resolving Ambiguity in Verbs

[Click here to watch the video](#)

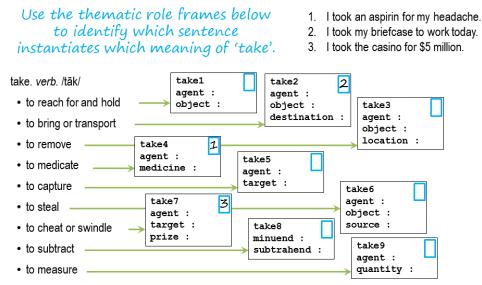


Figure 541: Exercise Resolving Ambiguity in Verbs

What did you decide on, David? So for the first sentence, my background knowledge tells me that the numbers 3 and 5 aren't objects. They aren't destinations, they aren't locations, they aren't targets. They're just numbers. Of all these frames, the only one that's looking for just numbers is the frame, to subtract. So I choose take8 as the meaning for the first sentence. So for the second sentence, my background knowledge tells me first that I'm an agent, that my briefcase is an object, and that work is a destination. So if I then try to map those three things to their various different categories here, I find that the only one that works is take2. I am an agent, my briefcase is an object, and work is a destination. So the third sentence a good bit more complicated. I still know that I am an agent, I know that the casino is a place of business, and the \$5 million could be considered a prize or an object. So that gets me down to take6 or take7. Take6 is to steal, take7 is to cheat or swindle, which are subtly different. But my background knowledge of the idea of source said that sources were usually indicated by the preposition, from. If I took \$5 million from the casino, then the casino would be a source. But I don't have from in front of the casino here, so I'm not going to regard it as a source. Instead I'm going to go with take7. That's an interesting analysis, David. Thank you. First of all, congratulations on becoming rich. I didn't know that you were going to become rich teaching this course. Of course, there are several other points to note here. One point is that you have a lot of background knowledge associated with the word for, perhaps because you are a native speaker of the English

language. Not everyone may associate the same kind of background knowledge with for and may not be able to disambiguate between these two. That actually gets to an underlying element of the problem as well, that these two meanings are themselves very similar. So it makes sense that it's more difficult to disambiguate between very similar meanings than disambiguating between stealing and medicating, for example. Another point to note here is that although we have been discussing about stories that are in the English language, this kind of analysis applies to all languages. Note also that so far we have been using sentences in the English language, but actually this kind of analysis works in respect to the language we are using. So I'm originally from India and my mother tongue is Hindi. And we can write the sentences in Hindi, and the same kind of analysis will work that is working here for the word take in English. This raises a few questions. Might this kind of analysis work for translation from one language to another language? If the different languages have similar kind of structures, then this analysis might help us translate sentences from one language to another language. Here's another question. Do all languages have the same kind of structure that enables mind to make use of the structure of the sentences to disambiguate between verbs? Feel free to take up that discussion over on our forums, especially if you yourself don't speak English as a first language. You might be particularly aware of the different structures present in English compared to whatever language you speak natively.

13 - The Earthquake Sentences

[Click here to watch the video](#)

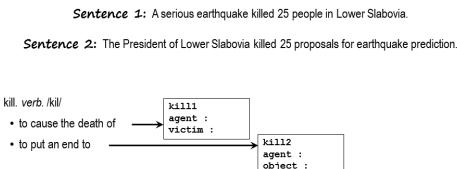


Figure 542: The Earthquake Sentences

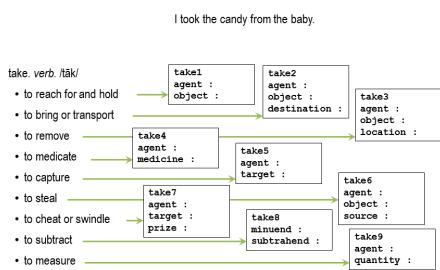


Figure 543: The Earthquake Sentences

So let us now return to our original example of these two stories and see how the analysis we have done, this semantic analysis, can help disambiguate between the two meanings of kill here. So, first my background knowledge tells me that kill can have several meanings, just like take had several meanings earlier. Kill can have the meaning of causing the death of someone, or kill can have the meaning of, to put an end to something. There could be other meanings of kill as well. Second, my background knowledge tells me that when kill has the meaning of, to cause the death of someone, it typically is a victim as well as an agent. In this particular case, the victim is 25 people, the agent is a serious earthquake. My background knowledge also tells me, that when the meaning to kill is to put an end to something, then typically it is both an agent that puts an end to something and an object that gets put an end to. In this particular case, 25 proposals is the object, and the agent is President of Lower Slabovia. It is this combination of background knowledge that allows me to infer the meaning of, kill, in the first sentence as, to cause the death of, someone. And kill, in the second sentence as, to put an end to some-

thing. I hope you can appreciate the power and beauty of this theory. But it is also important to point out that this theory has many limitations. To understand some of the limitations of this theory, let's go back to the sentence, I took the candy from the baby. In this sentence, we inferred that took signifies stealing the candy from the baby. And in fact, we had a large number of rules that told us how to make sense of the word take by making sense of the word candy, making sense of the word from. But as we look at increasingly large number of forms of the sentence, the number of rules that we need starts exploding. So consider small variations of the sentence. I took the candy for the baby. I took the toy from the baby. I took the medicine from the baby. I took the smile from the baby. I took a smile for the baby. They're all valid English language sentences, and each one of them tells a story. As I look at more and more variations of this sentence, I'll need to find more and more rules that can disambiguate between different interpretations of cake in those variations of sentence. In practice it turns out that it's very hard to enumerate all the rules for all the variations of sentences like this one. Nevertheless, the story appears to cover a good percentage of stories that we routinely deal with.

14 - Assignment Understanding

[Click here to watch the video](#)

Assignment

How would you use configuration to design an agent that could answer Raven's progressive matrices?

Figure 544: Assignment Understanding

So how would you use the notion of understanding [thematic] frames, constraint, and ambiguity to address Raven's progressive matrices? One example of an application here, would be to the idea of multiple possible transformations,

that we saw in some of our earlier problems. We saw certain problems that could be solved with either rotation or a reflection, but they would give different answers. You might imagine a frame that dictates certain expected values for different transformations. And the degree of fit to those expectations can dictate the accuracy of that particular answer. Try to think of the different phases of the understanding process. How do you first understand what's happening in the problem? How do you fit that, into a thematically role frame representation? And how would that representation then help you transfer and solve the problem?

15 - Wrap Up

[Click here to watch the video](#)

To recap...

- Thematic role systems
- Resolving ambiguity
- Leveraging constraints

Figure 545: Wrap Up

So today we've talking about understanding, which how agents make sense of stories, events and other things in the world around them. We started off by creating a more formal type of frame representation called thematic role systems that captures verbs and tell us what to expect in certain events. We then talked about how single verbs can actually have ambiguous meanings. But thematic role frames can help us differentiate which meaning a verb has in a particular sentence. Finally we talked about constraints and how certain words or frames can constrain the possible meanings of a sentence and help us figure out those ambiguous meanings. Today we've largely been talking about how single words or phrases can have multiple possible meanings, but next time we'll do this in reverse. We'll talk about how multiple words or multiple phrases or multiple sentences can actually have the same meaning. We'll talk about

how we can discern that sameness, and then react accordingly

16 - The Cognitive Connection

[Click here to watch the video](#)

In the lesson today, all of our examples came from natural language understanding. But understanding is a very journal purpose cognitive task. Natural language understanding is just one instance of understanding. Understanding is about making sense of the world. The world bombards us with data that comes in many forms. Acoustic, visual, verbal, numerical. It's a very hard problem. How do we make sense of the world? There are three sources of power. First, we exploit constraints about the world. We know that the world behaves in certain ways. Whether it's a physical world or the social world, or grammatical world. Second, we have structured knowledge representations. The structured knowledge representations in memory capture not just knowledge and its representation, but the organization of knowledge, and there is power in that organization. The power lies in the third part. Low level problem with processing helps us activate these knowledge structures from memory. Once activated, these knowledge structures generate expectations that make the processing top down. And there's a lot of power in being able to generate those expectations.

17 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

18 - Final Quiz

[Click here to watch the video](#)

Great. Thank you very much.

Summary

Understanding is the framework used for common-sense reasoning. Thematic role systems are the most structured type of frame representation and help resolve ambiguity in understanding the world. Agents make sense of stories, events and other things in the world

around them using Understanding. The Thematic Role Systems captures verbs and tell us what to expect in certain events and also handle conflicts and thus disambiguate meaning. Constraints and certain words or frames help constrain the meanings of a sentence and help us figure out ambiguous meanings.

References

1. Winston P., Artificial Intelligence,

Pages 249-266.

Optional Reading:

1. Winston Chapter 10, pages 209-220; [Click here](#)

Exercises

None.

Lesson 15 - Commonsense Reasoning



Believe nothing, no matter where you read it or who has said it, not even if I have said it, unless it agrees with your own reason and your own common sense.
– Buddha.

Common sense is not so common.
– Voltaire.

Inferences of Science and Common Sense differ from those of deductive logic and mathematics in a very important respect, namely, when the premises are true and the reasoning correct, the conclusion is only probable.
– Bertrand Russell.

01 - Preview

[Click here to watch the video](#)

Lesson Preview

- Primitive actions
- Actions and subactions
- State changes

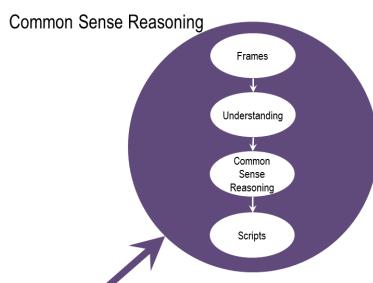


Figure 546: Preview

Figure 547: Preview

Today we'll talk about common sense reasoning. You and I, as cognitive individuals, are very good at common sense reasoning. We can make natural, obvious inferences about the world. How can we help AI agents make similar common sensical inferences about the world? Suppose I had a robot, and I asked a robot, find the weather outside. I don't want the robot to jump out of the window to see whether it's raining outside but, why should the robot not jump out of the window? What tells it that it's not a very common sensical thing to do? We'll talk about common sense reasoning using a frame

representation. We'll start talking about certain small set of primitive actions. There are only 14 of them, but they bring a lot of power because they organize a lot of knowledge. These primitive actions can be organized into hierarchies of sub actions. These actions can result in changes in the state. This is important, because that is what allows the robot to infer that if I were to take this action, that result might occur. That state might be achieved. So then it decides not to jump out of the window, because it might get hurt.

02 - Example Ashok Ate a Frog

[Click here to watch the video](#)

Ashok ate a frog.
 Ashok ate out at a restaurant.
 I could tell something was really eating him.
 The manager would rather eat the losses than devote more resources to the project.
 The river gradually ate away at the riverbank.
 The drip ate a hole through the wallpaper.
 The forest was completely eaten by the fire.
 He ate crow when his prediction was false.

Figure 548: Example Ashok Ate a Frog

Ashok ate a frog.
 Ashok ate out at a restaurant.
 I could tell something was really eating him.
 The manager would rather eat the losses than devote more resources to the project.
 The river gradually ate away at the riverbank.
 The drip ate a hole through the wallpaper.
 The forest was completely eaten by the fire.
 He ate crow when his prediction was false.

Ashok ate a frog.
 Ashok devoured a frog.
 Ashok consumed a frog.
 Ashok ingested a frog.
 Ashok partook of a frog.
 Ashok swallowed a frog.
 Ashok dined on a frog.
 Ashok inhaled a frog.

Figure 549: Example Ashok Ate a Frog

Okay. Have you ever wondered how you could write the equivalent of a Watson program or a CD program for your own computer? Just imagine if you could talk to your machine in terms of stories. Simple stories. Perhaps just one sentence stories like Ashok ate a frog. Now, we've already seen how a machine can make sense of this particular sentence, Ashok ate a frog. We did that last time. But, of course, eat or ate can occur in many different ways in

sentences. Here are some of the other ways in which I can use the verb eat. Ashok ate out at a restaurant. I could tell something was really eating him. And, the sense of eating him here, is very different from the sense of eating here. There's no physical object that is being eaten here. The manager would rather eat the losses. So now this is a very different notion of eat. The river gradually ate away at the riverbank. Yet another notion of eat. So just like we had in the previous lesson, take, the word take, which had so many different meanings, eat has so many different meanings. Now when we were discussing the word take, we discussed how we can use both the structure of sentences as well as background knowledge to disambiguate between different interpretations of take. We could do something similar with eat. We could enumerate all the different meanings of eat, then for each meaning of eat, we could ask ourselves what is it about the structure of the sentence and what is it about the background knowledge I have about things like rivers and river banks which tells me what the meaning of ate here is. To summarize this part, if you were to start talking to your machine in stories, in terms of simple stories designated by a single sentence, then one problem that will occur is, that words like ate or take will have large number of meanings. And we have seen how your machine can be programmed in such a way so as to disambiguate between different kinds of meanings of the same word. Now here is a different problem that occurs. Consider a number of stories again. Each story here is designated by a single sentence. Ashok ate a frog, Ashok devoured a frog, Ashok consumed a frog, Ashok ingested a frog, and so on. Several sentences here. And if we think about it a little bit, each of these sentences is using a different verb. But the meaning of each of these words in this sentence is pretty much the same. So whether Ashok ingested a frog or Ashok devoured a frog or Ashok dined on a frog, exactly the same thing happened in each case. There was a frog that was essentially outside of Ashok's body and then Ashok ate it up and at the end the frog was inside Ashok's body. The frog was dead and Ashok was happy. So the next challenge becomes, how

can a machine understand that the meaning of each of these words here is exactly the same? In a way this problem is the inverse of the problem that we had here. Here the problem was that we had a single word like eat, which had got a lot of different meanings and different context in different sentences. Here, the issue is that we have a lot of different words, but they have the same meaning given the context of the sentences. So another question then becomes, well how can we make machines understand that the meaning of these words in the sentences is exactly the same? Each of the sentences telling us exactly the same story. There is one other thing that is important here, and that is the notion of context. One of the hardest things in AI is, how do we bring context into account? In both of these cases, context is playing an important role. On the left side, context is playing an important role because we understand that the meaning of eat is different in these different sentences, because the context of eat is different. Here context is playing a different role. We understand that the meaning of each of these words is practically the same because the context here is practically the same. A couple of quick qualifications here. First, right now we're dealing with stories that are just one sentences. Very soon in the next lesson, we'll deal with stories which are much more complicated, which really have a series of sentences. For now, just simple stories. The second is that here is structural all of these sentences is the same. So structure practically tells us something about the context. But a situation is a lot more complicated because often two sentences, which have very different kind of structures can still have the same meaning. So consider these two studies. Bob shot Bill, and Bob killed Bill with a gun. Now here the sentence structures are very different, but their interpretations, their meaning are the same. Bill was the one who got killed. Bob was the one who did the killing. And the killing was done by putting a bullet into Bill. So the question now becomes, how can we build AI agents that can meaningfully understand stories like this one? And stories are the kind that Bob shot Bill, and Bob killed Bill with a gun. One thing we could do is, that for each of the

verbs here we could have a frame like we had a frame for take. So we could have a frame for ate, a frame for devoured, a frame for consumed, a frame for ingested, and so on. Well, we would have a lot of frames then. Because there are a large number of verbs in the English language or in any other modern language. Perhaps we could do something different. Perhaps we could look that there is a similarity between the interpretation of these things. Perhaps we could use the same primitive action for each one of them. So when we talk in English language, we might use for communication purposes all of these verbs. But perhaps we could compare AI agents that have a single internal representation for each one of them. What might that internal representation look like? We will call that representation a primitive action. Each one of these is an action. But many of these actions are equal in terms of our interpretation of those actions. Let's see what these primitive actions might look like.

03 - Primitive Actions

[Click here to watch the video](#)

Primitive Actions	
Move-body-part	Move-object
Expel	Ingest
Propel	Speak
See	Heart
Smell	Feel
Move-possession	Move-concept
Think-about	Conclude

Figure 550: Primitive Actions

Ashok ate ingested a frog.
Ashok devoured ingested a frog.
Ashok consumed ingested a frog.
Ashok ingested a frog.
Ashok partook of ingested a frog.
Ashok swallowed ingested a frog.
Ashok dined on ingested a frog.
Ashok inhaled ingested a frog.

Figure 551: Primitive Actions

David, that's a good point. One way of capturing the meaning of that story in terms of these

primitive actions is in, to use the move object primitive action. So, here about the object being moved and the mover of the object is the same, I. So, I am moving myself one place to another place. And the nearest point here is that these primitive actions will be able to capture the meaning of some sentences in a very simple, logical, intuitive sense. And for some other sentences, it might be a little bit more involved. So let us see how this primitive actions may actually help an AI agent make sense of stories. So here is the first thing we can do. Recall there were large number of stories, each expressed by one sentence. Ashok ate a frog, Ashok devoured a frog and so on. So now, ate, devoured, consumed, ingest, partook. We can think that each one of them really is an instance of a primitive action called ingest. Here's the primitive action Ingest. So Ashok ingested a frog here, Ashok ingested a frog here, Ashok ingested a frog here. Well, superficially these particular sentences are different. In terms of their deep meaning, the deep meaning is pretty much the same. But it's true of course that the world may have a slightly different interpretation than dining. The [verb] might be something that I ravish with my hands for example, and dining might involve fine dining with silverware. nevertheless, ingest captures the basic meaning of these sentences. What is the basic meaning? The basic meaning again is that Ashok is an agent. Frog is the object that is getting eaten, ingested. Initially the frog was outside Ashok's body, and probably dead or alive. We don't know its state. After the ingestion has occurred, a devoured has occurred, a consumed has occurred, the frog is inside Ashok's body, and it's dead. And further, that Ashok probably is happy at the end of this particular ingestion. And where is all of that knowledge coming from? It's buried inside the frame for ingest. So each of these primitive actions has a frame corresponding to it. And we have come across frame many times already in the class, so by now you should know what it means. And the frame has a large number of slots like agent, co-agent, object, beneficiary and so on, and those slots deal with still a difficult situations and have default values already put in there. So when a sentence

comes like, Ashok ate a frog, then the verb ate here, pulls out the primitive action ingest, and the frame for this primitive action and the processing becomes top down, and they start thinking, where will Ashok go? Where will frog go? Who is the agent? Who is the object? And so on.

04 - Exercise Primitive Actions

[Click here to watch the video](#)

Primitive Actions		Write the primitive action that subsumes the verbs in the following sentences.	
Move-body-part	Move-object	John	ran
Expel	Ingest	John	pushed
Propel	Speak	John	took
See	Heart	John	ate
Smell	Feel		
Move-possession	Move-concept		
Think-about	Conclude		

John ran.
John pushed the cart.
John took the book from Mary.
John ate ice cream with a spoon.

Figure 552: Exercise Primitive Actions

Okay, I hope you're enjoying this particular lesson, because I certainly am. Let's see, whether you're also understanding this lesson. So, here are four sentences, John pushed the cart, John took the book from Mary, John ate ice cream with a spoon, John decided to go to the store. And here's some of the words, are in these blue boxes. For each of the sentence, find the primitive action, that would best capture, the meaning of the verb inside the blue box.

05 - Exercise Primitive Actions

[Click here to watch the video](#)

Primitive Actions		Write the primitive action that subsumes the verbs in the following sentences.	
Move-body-part	Move-object	John	ran
Expel	Ingest	John	pushed
Propel	Speak	John	took
See	Heart	John	ate
Smell	Feel		
Move-possession	Move-concept		
Think-about	Conclude		

John ran.
John pushed the cart.
John took the book from Mary.
John ate ice cream with a spoon.

Figure 553: Exercise Primitive Actions

about. That sounds right David. There are sort of several other things to note here. What

is the last sentence that has two verbs in it? Decided and go. You'll see in a minute, how we can deal with sentences or stories that have multiple verbs in them. Second, you are right in that it is not always very easy to decide which is the best primitive action for capturing the meaning of a particular verb here. So the verb here was pushed, and the reason David propelled or moved object is because in propulsion the body is in some sense in contact with the particular object that is getting moved. Now we are not sure inherently detailed specification of each of one of these primitive actions. But I can tell you that the readings at the end do give them in detail

06 - Thematic Roles and Primitive Actions

[Click here to watch the video](#)

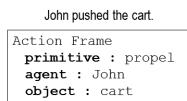


Figure 554: Thematic Roles and Primitive Actions

So let's dig deeper into, how the agent may do the processing using this primitive action. So, consider the sentence John pushed the cart. The AI agent beings left to right, because we're talking about English language here. He first word here John is not a verb. So, the AI agent for the time being, puts this in a list of concepts. And ignores it it comes to the second word in the sentence. Which is pushed, there's a word here, pushed, and now the AI agent uses this particular verb, push, as a probe into it's longterm memory. So now, the frame for propel is going to get pulled out, let's see what the frame would look like. So, here is the frame for propel that has been pulled out of the longterm memory? And this frame tells us that we can expect an agent. We can expect an object. And indeed, although

we are not shown here, this frame may have additional slots and perhaps additional things can be expected. Now for each of the slots, there is a rule budding in here. A rule which tells us, how can we pull out from a sentence, the entity that goes under the slot? The filler that must go here. So here's a rule that says that. If there is a concept just before the word, and that concept is animate, then, whatever that concept is, put it here. Well, there is a concept just before push, that's John, and let's suppose we have a lexicon which tells us if John is animate, then we put John here. Similarly there is a rule here for this slot, which tells the agent that if there is a object, there is a concept that comes after this verb. And that particular concept refers to an object that is inanimate. Then, take that particular thing, and put it here, and that's the cart. And so, we put cart here. So this way, this particular scheme, helps us derive the meaning of, John pushed the cart. And notice that the processing is a combination of bottom up and top down, as we discussed earlier. It begins, bottom up. Because we are looking at the data. Right now we don't have knowledge. But, as soon as some data is processed, it pulls in knowledge from memory. And soon the processing becomes top down, this frame helps to general expectations. And help pull things out. This is almost acting, like a hook for a fish. So once you have the hook, then you can capture the fish well. There are several things to be noted here. First representations like this. First, representations like this, are called structure knowledge representations. There's not only a representation here, but, there is a strong structure to it. Earlier when we were discussing predicates and logic, or we discussing rules and antecedents and consequence of rules. They are like the atoms of knowledge representation. They don't have much structure. But by now, we have this molecule of forms knowledge representation. And which a large number of atoms are getting connected with each other. And this connections are important, because once you have that structure of the molecule it tells you what can go in the place of each atom. Second, this is a simple sentence, and so the processing was quite simple.

The sentences need not always be as simple as this one. What happened to the sentence had the word, push in it? And I picked a frame, that is not the right frame for it. Suppose I had pulled out a frame for move object is shown as propel here. Well, one possibility is, that if you were to select a different frame for making sense of this particular sentence, there's a high I try to fill the slot for this particular frame, I will find a lot of difficulty. In which case, I might abandon the frame and try a different one. The second possibility is, that I may even force the interpretation of this into the slots for the other frame. But, if there is a study here which contains large number of sentences then, soon I'll realize this is not the right frame, I may abandon it and pick different frame. So for complex sentences, this processing is not as linear as we have pretended it to be right here. It is also possible, that sometimes one of the words en-map into two frames equally well. Indeed, this is the basis of many of the [puns] we encounter. So consider, I was wondering why the ball was becoming bigger. Then it hit me. Now, you of course understand the word hit there, that particular word, maps into two different interpretations, and that's why it is interesting and funny. So here's another one. Two men walk to the bar to, the third one ducks. Now here the word, the bar is overloaded. So here, walked into the bar, is getting interpreted two different ways. Indeed, people have tried to build accounts of humor, based on the kinds of story interpretation that we are doing here. Could it be, that this is beginning of a theory of humor. Of how you can tell stories not only to your machines?

07 - Exercise Roles Primitive Actions
[Click here to watch the video](#)

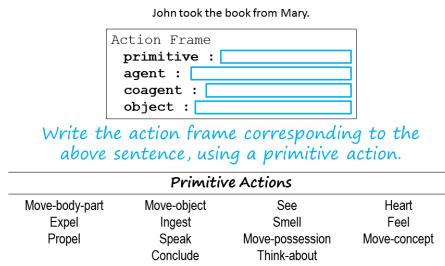


Figure 555: Exercise Roles Primitive Actions

Let's do an exercise together. Here is a sentence. John took the book from Mary. For this sentence, first write down the primitive action that best captures the meaning of the sentence. And then write down the fillers for each of the slots of this primitive action.

08 - Exercise Roles Primitive Actions
[Click here to watch the video](#)

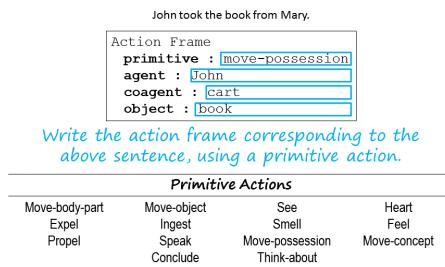


Figure 556: Exercise Roles Primitive Actions

That's good, David. Notice how lexical syntactical semantical analysis are all coming together here. The driver does semantic analysis. The driver does this frame which captures the semantics of this particular sentence that allows us to draw inferences about it. That's why we are use the term semantics here. So the semantics has been captured by this frame with terms of all of these slots. But how do we decide on the fillers? That requires lexical analysis and syntactic analysis. So, when we have the word John here, John is a concept and a concept is inanimate and that information is coming from a lexicon. That is the lexicon analysis. And when have a sentence like: John took the book from Mary. And from is a preposition

and Mary follows immediately after that. This is capturing part of this syntax of this particular sentence. And that is how we derive that the source must be Mary. So the semantic syntactical analysis is all working together here. Notice also how frames and routes are coming together here. You've seen how frames help us understand the meanings of the stories and that is being done part by the rules that are budded inside the slots. So there is a rule here which tells us how to extract the agent from the sentence and put it inside the slot. Similarly, there's a particular rule here and a rule here and each one of these slots may have its own rule. Of course as the sentences become complex and these frames become complex, these rules will become much more complex and sometimes there will be multiple rules here and multiple rules here. And this can become very complicated, very soon. Another point to notice here is that this capturing the semantics like I said earlier and how do we know that. Because I can ask questions. Who has the book? Who had the book? What did John take from Mary? And you can answer any of these questions using this frame. Once you have this frame, question answering becomes possible. Common sense reasoning becomes possible.

09 - Implied Actions

[Click here to watch the video](#)

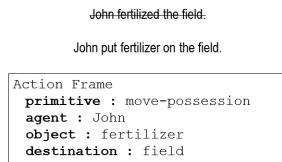


Figure 557: Implied Actions

The relationship between the structure of these sentences, and background knowledge is very intricate, and complex, and very entrusting as well. Sometimes the structure of sentences can be used to tell stories, have only implied actions in them. Consider the sentence, John

fertilized the field. Now it's hard to see fertilized mapping, into any of the primitive actions that we had here. There is an implied action here, that is not specified in the structure, the sentence here. But, it is much more meaningful in terms of the background knowledge that we have of those primitive actions. What John fertilized the field really means, is that John put the fertilizer on the field. Now put is something that maps in the primitive actions more easily. The response under the feature that's processing. Initially the inner agent must talk for the word, given in the sentence. And try to map in the primitive actions. And if that fails, then the [AI agent] agent must start thinking in terms of how to transform on the sentences, to bring out implied actions, that can more easily map the independent actions. This, again, is common sense reasoning. Through common sense reasoning, I'm interpreting that there must be implied action here, that captures the meaning of the sentence. Once I have made the implied action here transparent. The the rest of the processing, easier. The put action, maps into the primitive action of move object, this frame is pulled out, and the rest of the slots can be filled in, like earlier.

10 - Exercise Implied Actions

[Click here to watch the video](#)

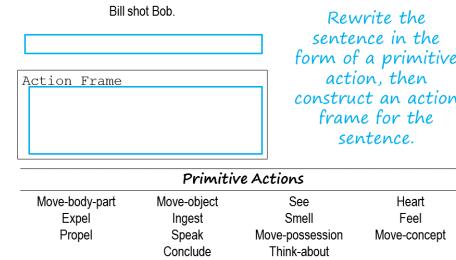


Figure 558: Exercise Implied Actions

Now let's do an exercise together. So consider the sentence, Bill shot Bob. Once again, shot is a verb here, that does not quite map internal [any of the] primitive actions clearly, cleanly. So perhaps there is an implied action here. So first write down the sentence in terms of a primi-

tive action, then write down the frame from this primitive action so fill the slots.

11 - Exercise Implied Actions

[Click here to watch the video](#)

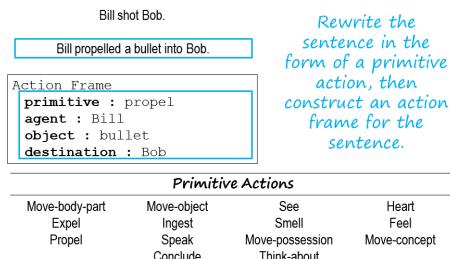


Figure 559: Exercise Implied Actions

What did you write, David? So, I wrote the sentence as Bill propelled a bullet into Bob. I looked up the verb shot and decided what that really means is pulling the trigger on a gun, which results in a bullet being propelled at high velocity out of the gun's barrel. So, it's fair to say that shot becomes propelled a bullet. Given that, I now have propel as my primitive. I have the same rule that says an animate agent that immediately precedes the verb becomes the agent in the frame. So, Bill is now my agent. I have my rule that says an inanimate object that immediately follows the verb becomes the object. So, bullet is now my object. Even though bullet wasn't in the original sentence, it's by rephrasing it that I can pull that object out. And I can use the constraints of into to determine that the destination of the bullet is inside Bob. That's good, David. But know this sentence could have another interpretation also. Imagine that Bill had a camera, and Bill will soon be taking pictures of Bob. That's true, Ashoke. And I bet if we had used Bill takes a picture of Bob as the meaning of shoot earlier in this lesson, that's probably what I would've thought of first. If I wanted to then put Bill shot Bob in terms of taking a picture into this form, I'd first rewrite it as Bill took a picture of Bob. That's really interesting, David. Because notice if you say Bill took a picture of Bob, it's not clear into what primitive action will this took map into. Perhaps we can discuss this

more on the forum. There's one more thing to notice here. Like David says, I can have multiple interpretations of Bill shot Bob. This sentence doesn't help me resolve between those interpretations. Perhaps it is something coming before this sentence in the story or something coming after that that will help me disambiguate. From the sentence itself we can simply construct two particular interpretations.

12 - Actions and Subactions

[Click here to watch the video](#)

Ashok put the wedge on the block.

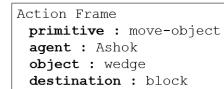


Figure 560: Actions and Subactions

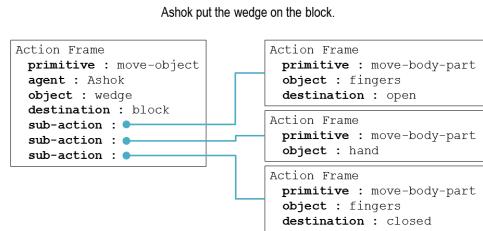


Figure 561: Actions and Subactions

13 - State Changes

[Click here to watch the video](#)

Ashok enjoyed eating the frog.

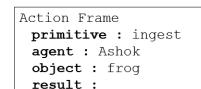


Figure 562: State Changes



Figure 563: State Changes

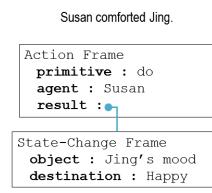


Figure 564: Implied Actions and State Changes

Nevertheless, let us continue and see how else this theory enables common sense reasoning. So consider the sentence, Ashok enjoy eating the frog. So not just Ashok ate the frog, Ashok enjoyed it. How do we make sense of this term? Once again, because the word eat, we may have the primitive action for ingest. Here is Ashok as the agent, frog as the object, and this time we'll have under the slot called result, which shows what happens when Ashok eats the frog. We will call this frame a state-change frame, where there was some initial state and some state at the end of this particular action, the action of ingest. So, here there is Ashok's mood, which is becoming happy. So Ashok, how would the agent know that in this case, the verb we're going to key on first is eating instead of enjoyed? Good point, David. So we can imagine a different action frame for feel because of enjoy being a feel. The agent as Ashok. Ashok is the one holding the feeling, and the object, or the feeling that is being felt, is joy. We can still have a frame for eating, and we can relate those two frames. So what specific interpretation an agent will make this particular sentence will depend on the precise rules that we're going to put into this slot so that the agent can make either this interpretation or that interpretation. We'll see another example of this situation in just a minute.

Sometimes it might not be clear to what exactly does this particular verb correspond to. So consider, Susan comforted Jing. Well, what exactly did Susan do to comfort Jing? It's not at all clear. Not clear what the primitive action should be. Although if you may not understand what exactly is the primitive action here. We want agents nevertheless to do common sense reasoning. Their interpretation might be incomplete, this interpretation might be partial. Nevertheless, you and I, as humans understand something about Susan and Jing here, that Susan for example, did something that made Jing happier. We want the agent to do the same kind of reasoning, without knowing what exactly is the comforting action here. So we may have a generic primitive action of do. This generic primitive action will use it, whenever we are unable to decide, whenever the agent is unable to decide, what exactly is the primitive action that should be pulled out. So the agent will simply say, well, Susan did something that made Jing's mood happy and this is as much interpretation that the agent might be able to the sentence, which is a pretty good interpretation.

14 - Implied Actions and State Changes
[Click here to watch the video](#)

15 - Actions and Resultant Actions
[Click here to watch the video](#)

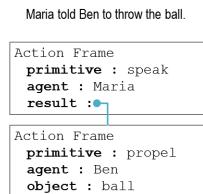


Figure 565: Actions and Resultant Actions

Earlier we had problems that will deal with sentences which have two verbs in it. So here are two verbs, told and throw. Maria told Ben to throw the ball. How may an AI agent make sense of this particular sentence? So once again, the processing starts on the left. Maria is not a verb, so let's put in a concept list and for the time being ignore it. The processing goes to the second word which is told, which is a verb. And so the primitive action calls when the told is pulled out. The primitive action is speak, and so now we can start putting the fillers for the various slots. So the agent is Maria and the result is now we go to the second one. Here the primitive action is propel because we have a throw here. And the propulsion is being done by Ben and the object is ball and now we relate these two. This second frame is a result of the first frame's action. So, if we go back to the previous sentence, Ashok enjoyed eating a frog. We can see how we can represent both verbs there in terms of action frames. So, Ashok enjoyed. That might be the frame here. The primitive action is feel. The agent is Ashok. Ashok ate a frog, that's the primitive action of ingest, agent is Ashok, and the object that got eaten was a frog. And the result of that is this frame where Ashok had a feeling of enjoyment. Note that some problems still remain. It is too difficult to figure out exactly how represent a sentence like Ashok enjoyed eating a frog. It can be two representations with that particular sentence and see that those are two action frames or one action frame and one state of change frame. Some of these questions will get answered when we stop thinking in terms of stories that are only one sentence long, but stories that have a number of sentences. Sto-

ries based on a discourse. Because some of these ambiguities will get resolved when we know more about what happened when Ashok enjoyed eating the frog. What came before it, or what came after it.

16 - Exercise State Changes

[Click here to watch the video](#)

Complete the frame representations for these sentences.

Anika decided to have a glass of water.

 Action ○ State Change

result : ●

 Action ○ State Change

result : ●

 Action ○ State Change

result : ●

Marc loved watching TED talks.

 Action ○ State Change

result : ●

 Action ○ State Change

result : ●

 Action ○ State Change

Primitive Actions

Move-body-part
Expel
PropelMove-object
Ingest
Speak
ConcludeSee
Smell
Move-possession
Think-aboutHeart
Feel
Move-concept

Figure 566: Exercise State Changes

So let's do a couple of exercises together. Here are two sentences at the top. Anika decided to have a glass of water and Marc loved watching TED talks. Please write down the action and the state change frames that will capture the meaning of the first sentence and the meaning of the second sentence.

17 - Exercise State Changes

[Click here to watch the video](#)

Complete the frame representations for these sentences.

Anika decided to have a glass of water.

 Action ○ State Change

primitive : conclude

agent : Anika

result : ●

 Action ○ State Change

primitive : ingest

agent : Anika

object : glass of water

Marc loved watching TED talks.

 Action ○ State Change

primitive : see

agent : Marc

object : TED talks

result : ●

Primitive Actions

Move-body-part
Expel
PropelMove-object
Ingest
Speak
ConcludeSee
Smell
Move-possession
Think-aboutHeart
Feel
Move-concept

Figure 567: Exercise State Changes

That's good David. Note though, that this sentence is a little bit like, a shark enjoyed eating the frog. This is one representation for this sentence, and under the representation we may have two action frames. One corresponding to the word loved, another corresponded to word watching, and then connect them, to the slot result. I hope you can see how agents might be able

to understand simple stories. In fact, this is quite similar to the way Watson and Siri go about to understand the stories that we talk to them. Almost surely the human interactions with the machines of tomorrow, will not be based on the keyboards and mouses that we have today. We'll talk to the machines, the machines will talk back to us, and when we talk to the machines we'll be telling the machines stories. Stories like Anika decided to have a glass of water, or a shark enjoyed eating the frog. And when we tell the stories, the stories will have context. They will have ambiguities. And we will expect the machines to do common sense reasoning. The power of this particular lesson lies in the [beginning] of a representation, that enables a particular kind of common sense reasoning. We'll continue this discussion, about common sense reasoning of more complex stories, in the next lesson.

18 - Assignment Common Sense Reasoning

[Click here to watch the video](#)

Assignment

How would you use commonsense reasoning to design an agent that could answer Raven's progressive matrices?

Figure 568: Assignment Common Sense Reasoning

So would you use commonsense reasoning to design an agent that could answer Raven's progressive matrices? Here you might make two connections. First you could connect primitive actions of agents, to primitive transformations in these matrices. Different problems could be composed out of a finite set of possible transformations. What would those primitive transformations be? And what would the action frames involved in each transformation look like? Second, you might connect those individual primitive actions to higher level transformations. What would your primitive transformations be? And

what common higher level transformations are possible? What primitive actions would result in those higher level transformations? And how would composing these like this, actually help you solve Raven's progressive matrices? In a way that you couldn't do otherwise?

19 - Wrap Up

[Click here to watch the video](#)

To recap...

- Primitive actions
- Actions and sub-actions
- State changes

Figure 569: Wrap Up

So today we've talked about common sense reasoning. Broadly, common sense reasoning gives us a formal structure to interpret the world around us. Having that formal structure let's us build agents that can then do the same. We started off with primitive actions, which like primitives in programming, are the simplest things we can interpret out the world. Then we looked at composing those primitive actions into bigger actions. For a hierarchical, abstract view of the world. We then looked at how those primitives can cause state changes, which let us predict the effect or cause of certain events. Next time, we're going to compose these simple frames into much longer stories, called scripts. Scripts help us make sense of complex repeated events with relative ease, and generate expectations about the world around us.

20 - The Cognitive Connection

[Click here to watch the video](#)

The connection between common-sense reasoning and human cognition is both very strong and not yet fully understood. Let us suppose that I were to ask you to go find the weather outside. You would not jump out of the window.

Why not? You would use common sense reasoning to know that jumping out of the window to find the weather is not a good idea. But what is it that tells you not to jump out of the window? You use the notion of goals. The notion of context to decide what not to do and what to do. We use similar notion of context in order to do natural language understanding. We could use context to disambiguate between various meanings of take. Can we context also to decide on what would be a good source of plan. So far we have been talking about common sense inferences about physical actions, what about the social world? You and I also make common sense inferences about the social world around us. One possibility is that you and I have a flurry of mind, this is actually called the flurry of mind theory. You and I as cognitive agents, ascribe goals, beliefs, desires to each other. And it's the theory of mind that allows us to make inferences about each other, including common sensical inferences.

21 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this

lesson.

22 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

Common-sense Reasoning gives us a formal structure to interpret the world around us and use it to build an hierarchical, abstract view of the world.

References

1. Winston P., Artificial Intelligence, Chapter 10, Pages 209-229.

Optional Reading:

1. Winston Chapter 10, pages 221-228; [Click here](#)

Exercises

None.

Lesson 16 - Scripts



Better the rudest work that tells a story or records a fact, than the richest without meaning.
– John Ruskin, *Seven Lamps of Architecture*.

To make a great film, you need three things – the scripts, the script, and the script.
– Alfred Hitchcock.

01 - Preview

[Click here to watch the video](#)

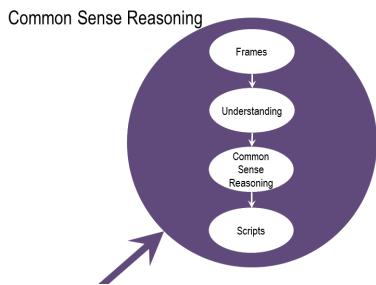


Figure 570: Preview

Scripts are the culmination of frames and understanding and common sense reasoning these last few lessons. We'll take what we learned about frames and understanding and common sense reasoning to build superior scripts. We'll start by defining scripts, then we'll talk about the form and the content of scripts. Then we'll discuss how we can use scripts to generate expectations about discourses and stories that help us make sense of the world around us. Finally, we'll talk about the hierarchical nature of scripts. I think you'll enjoy this lesson.

02 - Exercise A Simple Conversation

[Click here to watch the video](#)

Lesson Preview

- Defining scripts
- Form vs. content
- Generating expectations
- Hierarchies of scripts

Ali asked, "Do you think we'll have many customers in the next half-hour?"

Does this story make sense?

Yes

Sarah replied, "Go ahead and grab your lunch."

No

Figure 571: Preview

Figure 572: Exercise A Simple Conversation

Today, we'll talk about another knowledge representation called scripts. Scripts allow us to make sense of discourses and stories and scenes.

To motivate our discussion of scripts, let us continue with our metaphor of machines with

whom we can talk in stories. Now the Watson program and the Siri program, normally they understand stories that are limited to one sentence. You can ask Watson a question, you can ask Siri a question, and those questions are one sentence questions. Similarly when Siri and Watson reply, they typically give their answers in one word or one sentence at most. The story plays a very important role in the life of quadrant division and we would expect AI agents that live among quadrant divisions also to be able to understand stories. And one of the common roles that story players that they enable more common sense reasoning. To see how stories enable common sense reasoning, consider two simple sentences. Imagine the first sentence was, Ali asked, do you think we'll have many customers in the next half hour? And the second sentence is, Sarah replied, go ahead and grab your lunch. So these are two sentences of a story. Does this story make sense to you?

03 - Exercise A Simple Conversation

[Click here to watch the video](#)

<p>Ali asked, "Do you think we'll have many customers in the next half-hour?"</p> <p>Sarah replied, "Go ahead and grab your lunch."</p>	<p>Does this story make sense?</p> <p><input checked="" type="radio"/> Yes</p> <p><input type="radio"/> No</p>
---	--

Figure 573: Exercise A Simple Conversation

What do you think, David, does it make sense to you? So, to me this story does make sense. I can imagine it as a story where Ali and Sarah are co-workers and Sarah has some kind of authority over Ali. Ali is looking around the store and saying that there aren't that many customers right now, so maybe now is a good time take my lunch. But if there's about to be a rush of customers, then now is not a good time because leaves Sarah there by herself. So Ali asks Sarah, do you think there are going to be many customers in the next half-hour? Sarah knows

that it's around lunch time and Sarah recalls maybe that Ali had commented about missing breakfast this morning. So, she can infer Ali's motivation behind asking if there's going to be many customers, and she says, kind of anticipates the question he really was asking and says, yeah go ahead and take your lunch. So, yeah, I can find a way for this story to make sense, even though it doesn't seem like a clear connection between the two sentences. That's good David. Clearly you have an active imagination. Notice how David introduced lots of inferences here which were not really part of the input. How did he do it? This is common sense reasoning. How did he infer that it's probably lunchtime and that Ali wants to have lunch or that Sarah has some kind of authority over Ali? We have come across stories like this earlier. Here is another example. Ashok wanted to become rich, he got a gun. Two sentences and you can, I'm sure, understand immediately what might be going on. You may surmise that Ashok, as a professor, is unlikely to become rich, even if he teaches this online Master's course. On the other hand, you may have some stereotypical story about how some people become rich. Well, there are banks, and robbing a bank may require guns. And so now you can relate Ashok getting a gun to trying to become rich. So you and I, as human beings, have little difficulty in understanding stories like this. You might be given two sentences like this and we can connect them, causally connect them, coherently connect them, even if sometimes we use very active imaginations to do so. How can we do this conjoining of sentences into a coherent story? Put another way, how can we make AI agent do it? What kind of knowledge should the AI agent have and what kind of inferences must it make in order to be able to connect these two sentences?

04 - Story Understanding for AI Agents

[Click here to watch the video](#)

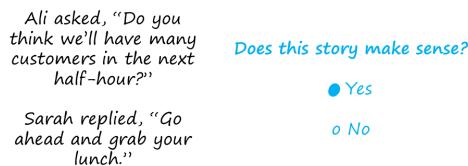


Figure 574: Story Understanding for AI Agents

That's a good story David. Now let's consider a different set of issues. Imagine that I told you a story. Bob went to a restaurant and sat down. But nobody came to serve him for quite awhile. Eventually, when someone did appear, he ordered a hamburger. The hamburger took a long time before it came. And when it did come, it was burned. Bob was not very happy. He didn't even finish the hamburger. Do you think Bob left a large tip? Well, I expect most of you would say no. He did not leave a large tip. If we had to wait for a long time. And if the food that came eve, eventually, was not of very high quality. You'd probably not even ask tip. But how did you come to that answer? Why did you expect, that in this particular case, Bob will not leave a large tip? Again, this connects to your notion of a story. It connects to your notion of a story, of what happens in a restaurant. Why do people leave tips? When do they leave them? To put it another way, stories help you make sense of the world. They help you [generate] expectations, even before [generate]. They allow us to make connections between events that otherwise might appear disparate. So in this lesson, we'll look at another structured knowledge representation called scripts. For representing stories or the kind that we are talking about. And we will see how this knowledge for presentation allows us to make sense of the world. And answer the kinds of questions that we are talking about.

05 - Visiting a Coffeehouse
[Click here to watch the video](#)

Notice I didn't need to tell David what coffee house, what time of day, what he was ordering, who the cashier was etc. He has a

script for that scene. And he associates it when needed. It helps with different expectations like, the cashier's going to give me my total, my drink should be coming up soon. If those expectations are not met, he knows something has gone wrong and he needs to react. This is the power of script. It helps to generate expectations about scenes in the world.

06 - Definition of Scripts

[Click here to watch the video](#)

Definition of Scripts
A causally¹ coherent² set of events³.
1: Each event sets off, or causes, the next event.
2: The causal connections between events make sense.
3: The parts are actions or scenes in the world.

Figure 575: Definition of Scripts

So what is a script? A script is a knowledge representation for capturing causally coherent set of events. Casually means that one event sets off another. So when David goes to the coffee house, as soon as he approaches the counter, the barista comes to him and says, what do you want? One event has set off the next one. Coherent means the links between these events make sense in the context of the world around us. So in David's script again, ordering coffee doesn't cause the barista to slap in the face. Because that would not be causally coherent in the context of the world. These events are referent to events in the world. Some events, like deciding or concluding, might be in the actor's mind, but for the most part these events are observable events.

07 - Parts of a Script
[Click here to watch the video](#)

Parts of a Script	
Entry conditions: Conditions necessary to execute the script.	Roles: Agents involved in the execution of the script.
Result: Conditions that will be true after the script has taken place.	Track: Variations or "subclasses" of the particular script.
Props: Objects involved in the execution of the script.	Scenes: The sequence of events that occurs during execution of the script.

Figure 576: Parts of a Script

So the structured knowledge for a presentation called script has six parts to it. The first part is called entry conditions. These are the conditions necessary to execute the script. So, for a restaurant script, as an example, the entry condition might be that there is a customer who is hungry and the customer has some money. The result refers to the conditions that will become true, after the script has occurred, after it has taken place. So, for restaurant script, the result might be that the owner of some restaurant has more money, the customer has less money, and the customer is now pleased and is no longer hungry. So the third part of a script is our props. So the third part of script are called props. Props are the kind of objects that are involved in the execution of this script so in case of the restaurant script the props might include tables and menus and food items and so on. So the fourth part of a script is roles. These are the agents involved in the execution of the script. As an example in the restaurant script it might be a customer who goes to a restaurant, the owner of the restaurant, the waiter or the waitresses in the restaurant, and so on. The fifth element of a script is a track. Track are variations or subclasses of a particular script. So for example in case of the restaurant script we may have tracks for going to a coffee house, or going to a fast food restaurant, or to a fine dining house. And finally the sixth element of a script refers to scenes. Scenes are specific sequence of events that occur during the execution of the script. So in case of the restaurant script there might be a scene for entering a restaurant. And a scene for ordering food. And a third scene for accepting the food and so on. When you put all of

the six elements together then you get a complete script. In the previous lesson we had taken the metaphor of a molecule as a knowledge representation. That is, knowledge representations that are not small or short or not atomic, but are molecule in nature, script is a big, large molecule.

08 - Constructing a Script

[Click here to watch the video](#)

Restaurant Script

```
Script
script : restaurant
track : formal dining
props : tables, menu, check,
money, F = food, P = place
roles : S = customer, W = waiter,
C = cook, M = cashier,
O = owner
entry : S is hungry, S has money
result : S has less money,
O has more money,
S is not hungry,
S is pleased
scenes :
```

Figure 577: Constructing a Script

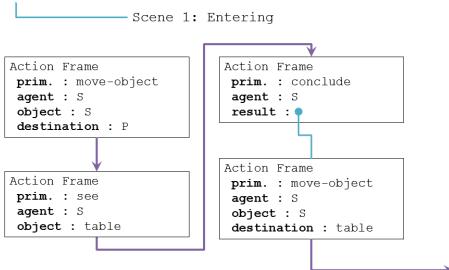


Figure 578: Constructing a Script

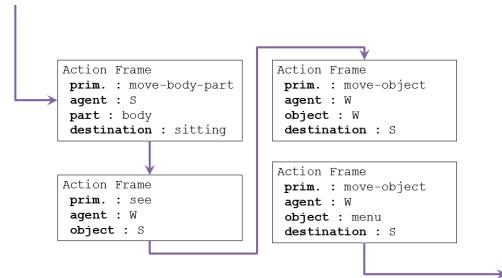


Figure 579: Constructing a Script

So here is a representation of the restaurant script. Here is the name of the script, restaurant, and the six elements that we talked about earlier. Track, props, roles, entry, result and

scenes. So this particular track refers to formal dining. Here are the props, tables, menu, check, money, food and place. And for food and place we can use symbols. These symbols can be used as variables. So where I may different kinds of foods and different kinds of places in a restaurant. Here are the rules, so S is a customer, W is a waiter, C is a cook, M is a cashier, O is an owner and so on. The entry conditions are S is hungry, S as the member is a customer, S has money and those conditions are that S has less money. S is not hungry. S is pleased. But O has more money. And scenes, well, let's discuss them in the next slide. Here is a representation of scene one. We'll call it the entering scene. This particular scene consists of several events. So in the first frame, the customer S, S stands for the customer. Moves himself or herself. So S is moving himself or herself to some restaurant, some place P. And the second frame, the agent S, the customer Sees some table. So this frame, the customer S decides to, take an action. The particular action is, where this agent S this customer S moves. Customer S himself or herself to the table. So this is a walking action going on. Let us continue another set just a little bit longer. So now S is moving his own body into a sitting position. Here, the waiter sees the customer, and now the waiter moves himself to the customer. And now the waiter moves the menu, to the customer. And this completes a representation of the first scene of entering in a restaurant. One can imagine many more scenes. The next scene might be a way to customer orders food. The third thing might be the way the waiter brings food, and so on and so forth. And then the last thing is the customer pays the bill and then walks out. This is a stereotypical notion of a script. Your notion of a script, might be slightly different depending on what kinds of restaurants that you go to. In different cultures, the script for going to a restaurant might be quite different. The point here is that the script is capturing in a knowledge representation, what is known about the stereotypical situation of going to a restaurant of a particular kind.

09 - Form vs Content

[Click here to watch the video](#)

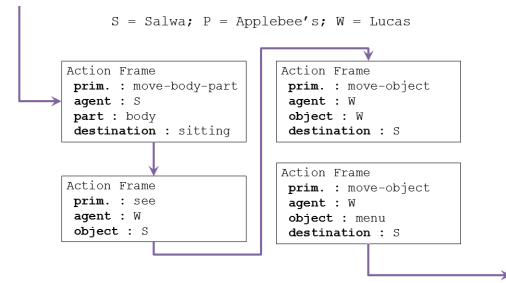


Figure 580: Form vs Content

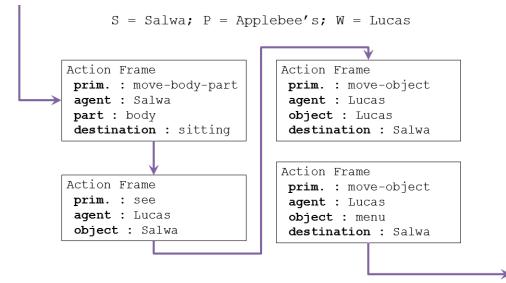


Figure 581: Form vs Content

So so far we have talked about a journal script of going to a restaurant, or an abstract script for going to a restaurant. This is like a class. We can instantiate it. So let's do this same script instantiated with these values with the greatest variables. So, Salwa is not a customer. Lucas is the waiter. And so on. And this instantiation is an important aspect of intelligence. Let's go back to our intelligent agent. It might be that Salwa is really a robot. Now, how would a robot know what to do in a restaurant? How do we program a robot in such a way that it would know what actions to take in a particular situation? Well, suppose that, Salwa the robot, in its memory, had a number of scripts like this one? And when it entered a restaurant, it invoked the restaurant script, which told it exactly what kind of actions to take. We could also see how this script allows Salwa, the robot, to generate expectations, to guess what will happen even before it happens. There is one more thing worth noting here. Notice how we

are composing scripts of these primitive actions here, the same primitive actions that occurred in the last lesson. So, these primitive actions are now providing the fundamental units of frames that compose together in some causally coherent sequence, make a script. This brings up another point. Notice how some knowledge structures are composed out of other knowledge structures. Earlier, we had frames for these primitive actions, gave them social knowledge structures. Now, we have scripts which are composed out of these framelike knowledge structures.

10 - Using a Script to Generate Expectations

[Click here to watch the video](#)

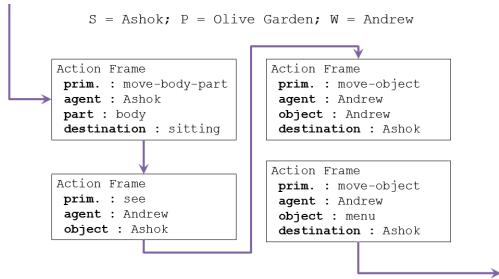


Figure 582: Using a Script to Generate Expectations

With effort to expectation generation just a while back let's look into it more deeply. So when we actually interact with the world sometimes things do not happen the way we expect them to happen. And sometimes things happen that people are not expecting. But how do we know what to expect and what not to expect? So consider a situation like this. This is an instance of a script. In this particular case, I have gone to Olive Garden. And the waiter is Andrew. So I move to a table and I sit down at the table. And now Andrew sees me. So from my perspective, I am expecting Andrew to come to me and give me a menu. But supposing this never happens. I am sitting there waiting for Andrew, or someone else, to come, and no one comes. That's an expectation violation. So I know something has gone awry. The reason I know that is because I continued expectations using the scripts,

expectations that are never fulfilled. Now consider an alternate situation. Instead of Andrew coming to my table and giving me a menu, Andrew comes to me and gives me a bill right away. Well I should think that I would be surprised by that. That was an unexpected event that occurred. Well several questions arise here. First, could this be the beginning of theory of surprise? In both cases, we were surprised. In one case where Andrew did not come and give a menu. In the second case, when Andrew came and gave a bill instead of giving the menu. On second notice, a surprise, it has two sides to it. One, when things do not happen the way we expect them to happen. And the other, when things happen the way in which we were not expecting them to happen. This example shows, that scripts not only tell us what to make sense of the world around us, they also tell us what to expect and what not to expect. And that's a powerful idea. So it seems like another example where this happens really strongly, that we encounter fairly frequently, is if you've ever seen a horror movie. One thing that horror movies really do really, really well is they lure you into expecting one thing to happen only to throw something completely different at you. So they lure you into kind of a false sense of security in expecting that nothing bad is going to happen in a particular scene, and then suddenly they throw a ghost at you from offscreen and it kind of startles you. And the reason it startles you is because it violates your expectations for what were going to happen next. David, that's a good point, that it happens in many different kinds of movies. So when I go and see a science fiction movie, or even a romance movie, I'm expecting certain things to happen. And sometimes I think a movie is really good if it is new and novel and different and it offers some surprising things. Notice that this could also be the beginning of a theory of creativity. In the last lesson we were talking about balance and humor. No we're talking about surprises. Some current theories of creativity say that a situation is creative if it is A, novel, B, if it is useful or valuable in some way, and C, if it is unexpected or surprising. Now, this begins to capture at least one of those three dimensions

of unexpectedness or surprise, and that's an important part. We'll return to computational creativity at the end of this course, when we'll talk a lot more about these issues.

11 - Tracks

[Click here to watch the video](#)

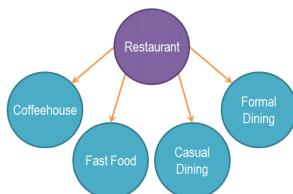


Figure 583: Tracks

Now, another part of the script was the track. And we really haven't talked a lot about track so far. So let's talk a little bit more about it. So here are four tracks with the restaurant script. That really [events occur] to going to restaurant, four kinds. Here's a coffeehouse, fast food, casual dining, formal dining. You could add more if you wanted to. Now in restaurants of all kinds, some even so common. You have to go to a restaurant, you have to order some food. You eat that food. You pay the bill. And then you leave. That is common to all of them which is why all of them are part of the restaurant script. On the other hand what happens in a Coffeehouse is quite different from what happens in Formal Dining, which is quite different from what happens in a Fast Food restaurant. So you may have specific cracks that correspond to Coffeehouses and Fast Foods and so on. In effect, we are building a semantic hierarchy or script. Here is a script for going to a restaurant. Here is a script for going to a coffeehouse, going to fast food. And this can be tracks in the overall script. Of course, we can build a semantic hierarchy of something higher than this. We could think about going to, for social events and [events occur] going to this restaurant becomes part of going to a social event of various kind. Okay now that we know

something about the [events occur] representation called script, the next question becomes how many AI agent actually use these scripts? So imagine an AI agent that is hungry has some money and decides to do something about it. So it may go into its long term memory and find out the script that will be most useful for the current situation. This really becomes a classification problem. In long term memory a large number of scripts, and the agent is trying to classify the current situation into one of those scripts. Let us suppose the agent picks a restaurant script, and decides to execute it. As it enters the restaurant, the scene it observes in the restaurant matches the conditions of a fast food script. So it decides to invoke the fast food script. This way the robot may walk down the semantic hierarchy, first in working the restaurant script, then working the fast food script, and so on. Now a robot could have taken a different stance. A robot could have decided to do planning. Given some initial conditions, and cold conditions, a robot may have used the operative that is available to it, to generate a plan at one time. While the script is doing it, it is giving it a plan in a compiled form. The robot doesn't have to generate this plan at runtime. It is already available in memory in a pre stored form. This is very useful because one of the central conundrums that we have been talking about is, how is it possible that AI agents can't address computationally complex problems with limited resources in near real time? In a complex dynamic world, planning can take a lot of time. But if I already have the store plans, then in working the script and executing it is much faster.

12 - Exercise Learning a Script

[Click here to watch the video](#)

Which of the following prior topics might help an agent learn a script? Check all that apply.

- Semantic Networks
- Frames
- Production Systems
- Learning by Recording Cases
- Incremental Concept Learning
- Planning
- Commonsense Reasoning

Figure 584: Exercise Learning a Script

So, we have talked a lot about how the notion of scripts is connected to many of the topics that we have discussed earlier in this course. So, let's do an exercise together. Which of the following topics might help an agent learn a script? Please check all that apply here.

13 - Exercise Learning a Script

[Click here to watch the video](#)

Which of the following prior topics might help an agent learn a script? Check all that apply.

- Semantic Networks
- Frames
- Production Systems
- Learning by Recording Cases
- Incremental Concept Learning
- Planning
- Commonsense Reasoning

Figure 585: Exercise Learning a Script

Which ones did you think, David, and why? So for learning a script, I said that five of the things we've talked about so far would really help an agent learn a script. So we've seen in the past that semantic networks and frames are representationally equivalent. We saw that when we put the raven's progressive matrices problems in terms of first semantic networks and then converted them to frames. Frames, as we've seen, are very useful for storing the type of information necessary to construct a thorough script. And if semantic networks are representationally equivalent, then we can also imagine a script composed of semantic networks instead. To skip the middle couple for a second, I can imagine incremental concept learning to be very important

to learning scripts. We can imagine an AI agent acting in the world and encountering multiple events everyday, and even to start to kind of develop a categorization scheme for those different experiences. So for example, that agent might learn that if I'm developing a script for fast food, whether or not I see a McDonald's logo or a Wendy's logo when I walk in, is not necessarily important to which script I run. But whether or not I see a counter with cashiers behind it or a hostess waiting to see me, is important. So that way, an agent can use incremental concept learning to learn the difference, for example, a fast food script and a fine dining script. So Ashok discussed before, planning happens when an agent has an initial state and a goal state and figures out how to navigate between the two. Once they figured out that plan for navigating between that initial state and that goal state, that then becomes a script that could be transferred to a new similar situation without having to completely re-plan the route from scratch. And finally common sense reasoning helps the agent out because it gives the agent a kind of a language within which to learn the script in the first place. It can learn a script within this language of primitive actions that it understands and then can use those to make sense of new and novel situations. Production systems and learning by recording cases don't really apply as much to scripts because they both involve representations at a very different level of abstraction, at a very low level of abstraction. With learning by recording cases, we tend to stick with the cases, whereas scripts we have an abstraction over them. And production systems are more like atoms of knowledge representation instead of molecules or compounds like we deal with with scripts. That's good, David. I may add, one of the things regarding the semantic networks. Recall that when we discussed semantic networks, we had considered how we could use semantic networks to interpret stories. We'll use this same example. Ashok wanted to become rich. He got a gun. And we said that inside a semantic network the notes that correspond to Ashok wanted rich and gun get activated. And the activation spread from there and there's a

path that formed a spare semantic network that path is the interpretation of this particular story. In a sense a script is that part. Of course if you think you see a connection between production systems or learning by recording cases and scripts that I haven't seen. Or if you think the connection between the other topics and scripts isn't quite as close as I've described, feel free to head over to our forums and we'll discuss it there.

14 - Exercise Using a Script

[Click here to watch the video](#)

Which of the following prior topics might help an agent use a script? Check all that apply.

- Generate & Test
- Means-Ends Analysis
- Problem Reduction
- Case-Based Reasoning
 - Classification
 - Logic
 - Understanding

Figure 586: Exercise Using a Script

Okay, let us do one more exercise together. This predicate exercise has to do with using a script rather than learning a script. Which of these topics that we have discussed earlier apply to an agent using a script? Check all that may apply.

15 - Exercise Using a Script

[Click here to watch the video](#)

Which of the following prior topics might help an agent use a script? Check all that apply.

- Generate & Test
- Means-Ends Analysis
- Problem Reduction
- Case-Based Reasoning
 - Classification
 - Logic
 - Understanding

Figure 587: Exercise Using a Script

David, which of these topics do you think are applied to using a script? So I chose four of the seven as applied to using a script. So for problem

reduction, we saw earlier that the script breaks down the overall scene into smaller scenes, and even further into smaller actions. What that means is that when we're executing the script, and it kind of gets caught somewhere we can break it down and see exactly where the script got caught. So we can see exactly where an expectation was violated. Classification we actually already discussed because classification can help us identify which script to execute in a given situation based on what we see. So if we walk into a restaurant and see a hostess for example we can classify that as a specific kind of restaurant and launch the script that goes along with it. Although it would take a bigger jump, I can also imagine putting a script in terms of formal logic. Especially because we discussed before, that a script can be considered a plan that has already been executed once, and can be transferred to new situations. So if we're discussing plans in the form of formal logic, we may also be able to put scripts in those same kind of terms. In terms of what they're asserting is true for a given state of the script. So if we could put plans in the form of formal logic, we can also imagine rewriting our script in the form of formal logic. That would give us a script in terms of what different elements of the script assert about the state of the world at different points of the script's execution. Finally, we can also see understanding applying pretty directly to scripts because it helps us disambiguate similar events in different situations. So to go with Ashok's example about receiving the bill right when you sit down at the table, understanding talks about how we can disambiguate that event based on what else has happened before and after. I didn't see the other three as being as applicable to scripts for a couple different reasons. For Generate & Test and Means-Ends Analysis, these are problem solving methods. And as we talked about with planning in the previous exercise, scripts are often used when we already have a solution and we simply need to execute it. Case-Based Reasoning keeps things at the level of individual cases that can be adapted to our current problem. Where as scripts serve as an abstraction over a number of cases. So I don't really see case-based

reasoning applying as much here either. This is good, David. Thank you for sharing this. Note that Generate and & Test, and Case-Based Reasoning might be able to have the ability to use scripts after all. So one can imagine a situation where there are a large number of scripts available and the robot has to decide which of the scripts should I use for a particular situation. And may not be able to classify the situation dashed into scripts and with that case the robot will pick a script, generate it, try it out, see if it works, if it does not pick another one. Also Case-Based Reasoning is currently the application of scripts in the sense that both Case-Based Reasoning and script-based reasoning are extremely memory intensive. What both of them are saying is that memory often supplies most of the answer. Like we said earlier when we were discussing Case-Based Reasoning, we don't think as much as we think we do. Most of the time memory gives us the answer. The difference, of course, like David pointed out is, that cases defer to instances whereas scripts are abstractions of the instances.

16 - Assignment Scripts

[Click here to watch the video](#)

Assignment

How would you use scripts to design an agent that could answer Raven's progressive matrices?

Figure 588: Assignment Scripts

So how would scripts help inform the design of an agent to solve Raven's progressive matrices? Remember, scripts are ways of making sense of complex events in the world and we can certainly consider individual Raven's matrices to be complex situations. You thus might have a script for different broad categories of Raven's problems. If this was your approach, what would your entry conditions be for each script? What

would the tracks be? What would the scenes be? Where are these scripts going to come from? Are you going to tell the agent what script it should use or will it learn a script from prior problems? If the agent succeeds using the script that you give it, who is intelligent? You or your agent?

17 - Wrap Up

[Click here to watch the video](#)

To recap...

- Defining scripts
- Form vs. content
- Generating expectations
- Tracks

Figure 589: Wrap Up

So today we've talked about scripts, a complex way of understanding stories in the natural world. Stories aren't just narratives, though. Painting, songs, buildings are all stories of different kinds. Stories are around us every single day. We started off by defining scripts. Scripts are causally coherent series of events. They give a prototype for what to expect in certain situations. The form of the general script shows us the form of the overall prototype for the situation. A specific instantiation of the script then specifies the content. Scripts can have different tracks as well. At a high level, any kind of restaurant involves entering, ordering, paying, and leaving. More narrowly though, fast food and drive through restaurants involve different scripts from casual or formal dining. This concludes our unit on common sense reasoning, but note that some of what we cover in the future will be applicable to learning, differentiating, and refining scripts.

18 - The Cognitive Connection

[Click here to watch the video](#)

Scripts are strongly connected to current theories of human cognition. In fact one recent theory says that brain is a predictable machine. We do very quick bottom up processing followed by

mostly top down processing with general expectations of the world. Then we act on those expectations. This idea in fact is so strong, that when it fails it leads to amusement, or surprise, or anger. If I violate the expectations of your script, you might find it funny or surprising or you might be upset about it. An interesting and open question is, whether we carry the scripts in our head or do we generate them at run time? Scripts are also current with the notion of mental models. You and I have mental models, or scripts. Not just about social situations like going to a restaurant, going to a movie, but also about how the computer program works, how the economy works, how the car engine works, your physical, social, economic works. Note that scripts can be culture specific. In the U.S. for example, going to a restaurant typically involves leaving a tip. But in many countries, this is not the case. In fact in some countries, tipping is considered insulting. So scripts presumably evolved through cultural interaction over long periods of time. But once there, they're a very powerful source of knowledge.

19 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

20 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

Scripts help us make sense of complex, repeated events with relative ease and generate expectations about the world around us.

Scripts are the culmination of frames and understanding and common-sense reasoning. Scripts are causally coherent series of events, that gives a prototype for what to expect in certain situations.

References

1. Winston P., Artificial Intelligence, Chapter 10, Pages 209-229.

Optional Reading:

1. Schank & Abelson, Scripts, Plans and Knowledge; [Click here](#)

Exercises

None.

Lesson 17 - Explanation-Based Learning



If you can't explain it simply to a six year old, you don't understand it yourself.
— Albert Einstein.

The longer the explanation, the bigger the lie.
— Chinese Proverb.

01 - Preview

[Click here to watch the video](#)

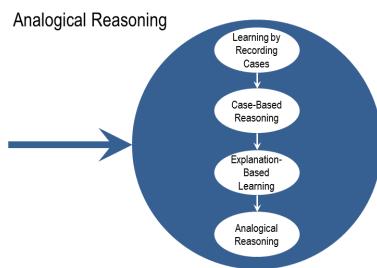


Figure 590: Preview

it learns about connections among existing concepts. We'll use explanation-based learning to introduce the notion of transfer of knowledge from an old situation to a new situation. This will help us set up infrastructure needed for talking about analogical reasoning next time. Today, we'll start by talking about a concept space where we will map out all the concept and relationships between them. Then we introduce the notion of abstraction that helps us do transfer. Finally, we'll use transfer to build complex explanations that will lead to explanation-based learning.

02 - Exercise Transporting Soup

[Click here to watch the video](#)

Lesson Preview

- Concept space
- Prior knowledge
- Abstraction
- Analogical transfer

Which of these might be useful for transporting soup from the kitchen to the table?

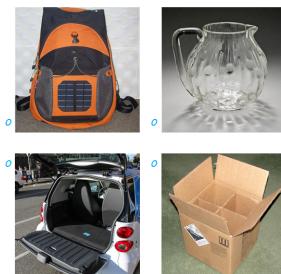


Figure 591: Preview

Figure 592: Exercise Transporting Soup

Today, we'll talk about explanation-based learning. In explanation-based learning, the agent doesn't learn about new concepts. Instead,

To illustrate explanation-based learning, let us begin with an exercise. Imagine that you want

to transport soup from the kitchen to the dining table. Unfortunately, all of the usual utensils you use to transport soup are unavailable. They're dirty, or just not there. So you look around, and you see some objects in the vicinity. Here is your backpack, here is a pitcher, here is a box, here is your car. And you wonder which one of these four objects could you use to transport soup from the kitchen to the dining table. Well, which one would you use?

03 - Exercise Transporting Soup

[Click here to watch the video](#)



Figure 593: Exercise Transporting Soup

I think most of us would give the same answer. The pitcher, not the backpack or the car or this box. Now for humans, for you and me, this is a little bit of an easy problem. All of us get it right pretty much all the time. And what about a machine? What about a robot? How would a robot decide that a pitcher is a good utensil to use for transporting soup from the kitchen to the dining table, but not a backpack and not a box? For a robot, this is a surprisingly hard problem. So the question then becomes, well, what is it that makes it easy for humans and hard for robots? How can we program AI agents so that it would be easy for them as well? One important thing to note here, this is another example of incremental learning. Now we have come across incremental learning earlier, when we were talking about incremental concept learning. There we were given one example at a time, that was the example of art, and we were learning one concept, the concept of art. This is in contrast to other methods of machine learning. Where one is given a, large amount of data,

and one has to detect patterns with a variety in that data. Here, there is learning occurring one step at a time, from a small number of examples, one single concept has been learned. We also came across the notion of incremental learning, when we were talking about chunking. Day two, there was one particular problem, and from a small number of previous episodes, we chunked a particular rule. This notion of incremental learning, for me it's much of knowledge based AI. Another thing to note here, this notion of expression based learning is related to creativity. We talked earlier about the relationship between creativity and novelty. Here is an example in which an AI agent is dealing with a novel situation. Usual utensils for taking soup from the kitchen to the dining table are not available. What should the robot do? The robot comes up with a creative solution of taking the pitcher as the utensil

04 - Example Retrieving a Cup

[Click here to watch the video](#)

A Cup
A cup is an object that is stable and enables drinking.

An Object
This object is light and made of porcelain. It has a decoration, a concavity, and a handle. The bottom is flat.

Can we prove this object is a cup?

Figure 594: Example Retrieving a Cup

So imagine that you have gone to the hardware store and bought a robot. This is a household robot. Usually the robot goes into the kitchen, makes coffee and brings it to you in a cup. However, last night you had a big party, and there's no clean cup available in the kitchen. The robot is a creative robot and looks around, and it finds an object. And this object is light and made of porcelain. It has a decoration. It is concave. It has a handle. The bottom is flat. The robot wonders, could I use this particular object like a cup? It would want to prove to itself, that this object, in fact, is an instance of this concept of cup. How might the robot do it?

05 - Concept Space

[Click here to watch the video](#)

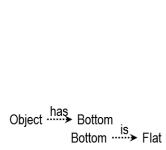


Figure 595: Concept Space

So let us now see how the AI agent may go about building an explanation. Why this particular object is an instance of this cup? So this what the robot wants to prove, the object is a cup. And this what the robot know about the object. Object has a bottom, the bottom is flat, the object is made of porcelain and so on and so forth. So the question then becomes, how might the AI agent embodied in the robot, go about using its prior knowledge, in order to be able to build this explanation? But another way, what prior knowledge should the AI agent have, so that it can, in fact, build this explanation?

Prior Knowledge

[06 - Prior Knowledge](#)
[Click here to watch the video](#)

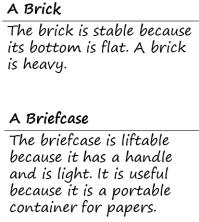


Figure 596: Prior Knowledge

A Brick
The brick is stable because its bottom is flat. A brick is heavy.

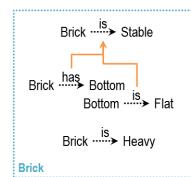


Figure 597: Prior Knowledge

A Briefcase
The briefcase is liftable because it has a handle and is light. It is useful because it is a portable container for papers.

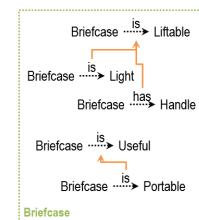


Figure 598: Prior Knowledge

A Bowl
The bowl carries liquids because it has a concavity. The bowl contains cherry soup.

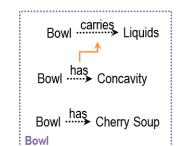


Figure 599: Prior Knowledge

A Glass
The glass enables drinking because it carries liquids and is liftable. It is pretty.

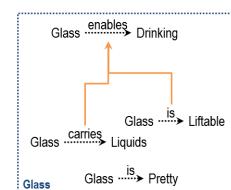


Figure 600: Prior Knowledge

Let us suppose that the A agent has prior knowledge of these four concepts. A brick, a glass, a bowl, and a briefcase. Let us look at

one of these concepts in detail. So the brick is stable because the bottom is flat. And a brick is heavy. So first, this particular conceptual characterization has several parts. About stability and about heaviness. Second, there is also something about causality here. The brick is stable because its bottom is flat. This is an important element of the brick, and similar things occur in the other four concepts. Let us now consider how an AI agent might be able to represent all of this knowledge about a brick. Here is a visual rendering of this representation. First, the A agent knows about a lot facts about the brick. So a brick is heavy. The brick has a bottom and the bottom is flat. That comes from the second part of the first sentence. Bottom is flat. And also that the brick is stable. So here are some observable facts and this is a property of the brick. So this is part of the structure of the brick. This is part of its function. In addition, the A agent knows that the brick is stable because the bottom is flat. So we need to capture this notion of causality. So this yellow arrows here, are intended to capture this notion of causality. The AI agent knows that the brick is stable because the brick has a bottom and the bottom is flat. And this way, it connects these structural features into these functional features, to these causal connections. To take another example, here is a conceptual characterization of a briefcase and here is its knowledge for representation. I'll not go through this in great detail, but briefly, a briefcase is liftable because it has a handle and it is light. And it is useful because it contains papers. Notice the notion of causality here again. Once again there're these facts about the briefcase, for example the briefcase is portable, the briefcase has a handle, you know, these structural observable features. And then there is this functional features like briefcase is useful and briefcase is liftable. And then we have these yellow arrows denoting the causal connections between them. Similarly for the bowl, here is its conceptual characterization and the knowledge representation. So the bowl contains cherry soup that one of the fact here. The bowl is concave, that's another fact here and it is a causal relationship here. It carries liquid because it is

concave. Finally the fourth concept the A agent knows about, a glass. And a glass enables drinking because it carries liquid and it is liftable, and it is pretty. So the glass is pretty. The glass carries liquids. It is liftable. And the fact that it enables drinking is because of these two other facts. Note quickly that not all these structural features participate in this causal explanation.

07 - Abstraction

[Click here to watch the video](#)

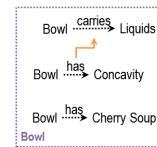


Figure 601: Abstraction

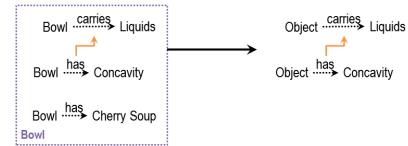


Figure 602: Abstraction

Now that we have the characterizations and the knowledge representations the four concept's worked out, let us see how the AI agent might actually use them. So let's look at the bowl. Here was the knowledge representation of the characterization of the bowl. The AI agent will abstract some knowledge from this particular example. Here is its abstraction. Two things have happened here. First, it is abstracting only those things, that are in fact causally related. Simple features that have no causal relationship with other things, are not important and they can be dropped. So we can add one other element of a notion of an explanation. The explanation is

a causal explanation. The AI agent is trying to build a causal explanation that will connect the instance, the object, into the cup. Second, the AI agent creates an abstraction of this characterization of the bowl. And so in the bowl, it replaces it with an object. So here the bowl carries liquids, because it is concave, and it is abstracted to the object carries liquid because it is concave. This is the abstraction that is going to play an important role in constructing the causal explanation.

08 - Transfer

[Click here to watch the video](#)

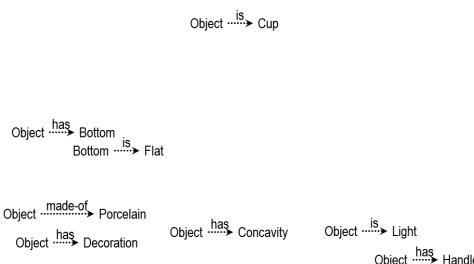


Figure 603: Transfer

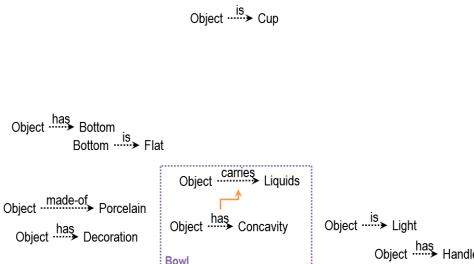


Figure 604: Transfer

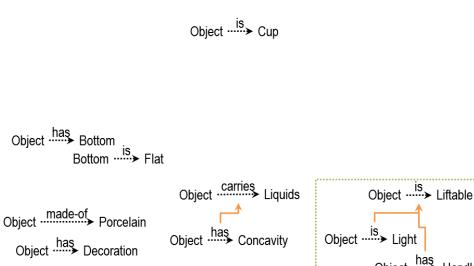


Figure 605: Transfer

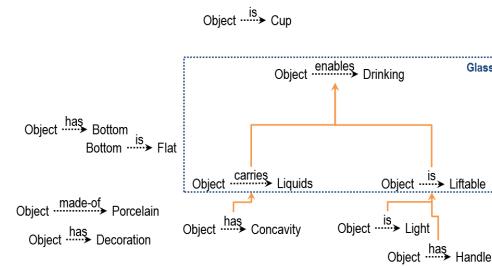


Figure 606: Transfer

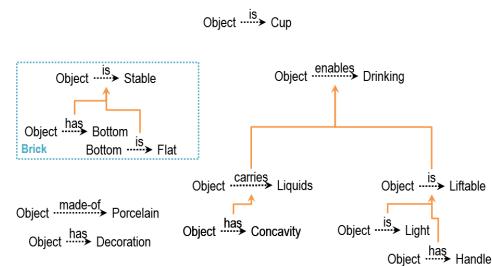


Figure 607: Transfer

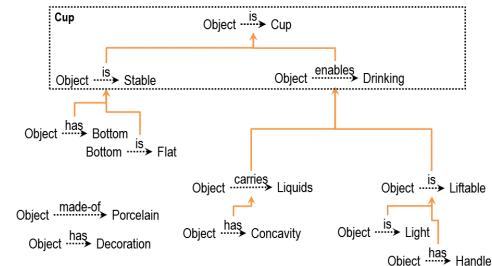


Figure 608: Transfer

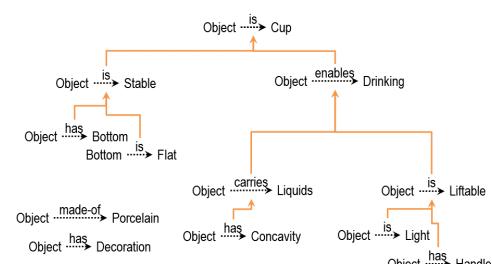


Figure 609: Transfer

Okay so now let us see how the AI agent might go about building an explanation that shows that the object that have these properties in fact is an instance of this concept of a

cup. What I'll do here is to first sketch out the explanation for you and then I'll discuss how the AI agent goes about building the explanation. Let us first consider this part. This is an abstraction that we learn from the bowl. The object carries liquid because it is concave. There is a similar abstraction learned from the briefcase. The object is liftable because it is light and it has a handle. Here's the abstraction that the air agent learned from the glass. The glass enables drinking because it carries liquids and because it is liftable. So, in this way, the abstraction learned from the glass gets connected to the abstractions learned from the bowl and the abstraction learned from the brief case. Note that the effect here and the abstraction from the bowl has become the cause for the effect that was the abstraction from the glass. Similarly here, the effected object is liftable was learned from the briefcase example, but in the context of the glass, this becomes the cause for object enables drinking. Let's shift to the fourth example for brick. From there, one knows that the object is stable because it is a bottom and the bottom is flat. Now, we can connect this causal chain and this causal chain into a more complex causal explanation for why this object is a cup. The call that we had said earlier and the definition of a cup, that an object is a cup if, it is stable and it enables drinking, now we have shown that in fact, this particular object is a cup because it is stable and it enables drinking. In practice when an A Agent actually goes about building this explanation, it works backwards. For the definition of the cup, it knows that in order for an object to be a cup, the object must be stable and the object must enable drinking. So the A Agent's senses' goal of proving that the object is stable into his memory. Memory returns the example of a brick, because a brick is stable, but the brick said that the brick is stable because it has a bottom and the bottom is flat. So the air agent affects this causal relationship from the brick example and applies it to the object here. Similarly for this part and they asked for the definition of the cup was that it enables drinking. So how can the A agent prove that the object enables drinking? This object only exists

in the kitchen. The air agent sensed the goal of proving that the object enables drinking induced memory. Memory returns the example for glass because the glass enables drinking. It abstracts from glass the fact that the object must carry a liquid and the object must be liftable in order to enable drinking. That sets up some new experiences with sub goals. How can the area agent now prove that the object can carry liquid? That goes back into the memory, and the example of a bowl is found. And I think you can now see how the rest of the explanation process would work. Note this connects with a couple of lessons we have learned earlier. First you can see [problem] reduction and application here. Wanting to prove that object is a cup, to do that we deduce the problem into two smaller and simpler problems. Let's prove that object is stable, let's prove that object enables drinking, and so on and so forth. Second this connects with the notion of planning. When we were discussing planning we had open preconditions. Those were preconditions of a goal which we needed to satisfy. We use those open preconditions to select operators, a similar thing is happening here. When we want to prove that the object is a cup, then we have two open preconditions, object is stable, and object enables drinking. These preconditions then become levels while selecting explanatory proofs, explanatory fragments that can help fulfill those preconditions. So the scenario of a robot proving that an object is a cup is a bit of a longer term scenario. We're still a ways away from having robots that walk around our house and retrieve cups for us. To take an example perhaps from the near future, I can imagine one day having a word processor that has built into it some intelligence such that, instead of choosing a file based strictly on its file name, I can say something like, get me that important document from last Tuesday. I myself might not remember the filename or where I saved it, but I remember that last Tuesday I was working on an important document. The intelligent search in the word processor would then iterate over the documents and say, can I both prove that the document is important and it is from last Tuesday? Now the word processor might not have a notion of im-

portance built into it naturally, but it knows documents I considered important in the past. And it might start to notice the documents that have my header on them, and have my signature at the bottom, are important. And from there, it can start to build a case for why that particular document is important. Coupled with just being able to read the time stamp and see if it's from last Tuesday, it should be able to retrieve for me documents that might be both important and from last Tuesday without me ever really knowing what the document file name was or where I saved it. This kind of explanation-based learning actually occurs in our everyday life. So you are now constantly improvising. Papers are blowing off my desk. How can I stop them from blowing off? So I need something to stop them from blowing away. What is other label what can act as a paper weight? A cup. Here is a cup. Let me put it on the paper. This is an example of improvisation where we use explanation-based learning to realize that this is how we use a flat bottom can act as a paper weight. Here is another example. I need to prop open a door, a door stopper is not available. What can I use? Perhaps an eraser or a chair. You and I do this kind of improvisation all the time, and often we are building these explanations that tell us that an eraser can be used as a door stopper.

09 - Exercise Explanation-Based Learning I

[Click here to watch the video](#)

A Mug A mug is an object that is stable, enables drinking, and protects against heat.	A Pot The pot carries liquids because it has a concavity. It limits heat transfer because it has thick sides and is made of clay.
An Object This object is light and made of clay. It has a concavity and a handle. The bottom is flat, and the sides are thick.	
Can we prove this object is a mug? <input type="radio"/> Yes <input type="radio"/> No	

Figure 610: Exercise Explanation-Based Learning I

Okay let us do an exercise together. This time instead of showing that an object is an instance of a cup, we are going to try to show that

an object is an instance of a Mug. So here is the definition of a Mug. A mug is an object that is stable, enables drinking, and protects against heat. Notice that we have added one more element here, not only stable like a cup, not only enables drinking like a cup, but also protects against heat. Here is an object, I will label in the cushion. The object is light and is made of clay. It has concavity and has a handle. The bottom's flat and the sides are thick. You can assume that the agent knows about all four examples as earlier, the glass, the bowl, the brick, and the briefcase. [In addition,] in this particular case the agent also knows about yet another example, a Pot. The Pot carries liquid because it is concave, It limits heat transfer because it has thick sides and is made of clay. Your task is to. Build an explanation that shows, that this object is an instance of a Mug. Can we prove this?

10 - Exercise Explanation-Based Learning I

[Click here to watch the video](#)

A Mug A mug is an object that is stable, enables drinking, and protects against heat.	A Pot The pot carries liquids because it has a concavity. It limits heat transfer because it has thick sides and is made of clay.
An Object This object is light and made of clay. It has a concavity and a handle. The bottom is flat, and the sides are thick.	

Can we prove this object is a mug?
 Yes No

Figure 611: Exercise Explanation-Based Learning I

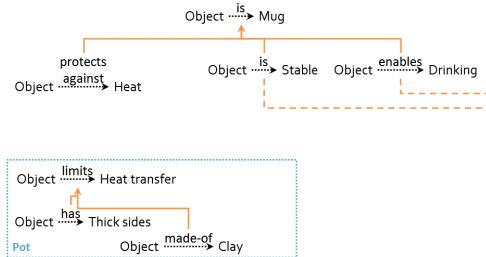


Figure 612: Exercise Explanation-Based Learning I

LESSON 17 - EXPLANATION-BASED LEARNING

What do you think, David? This is provable. Some said that no, based on the knowledge that we're given here about the mug, a pot, and an object, and the things that we know from the past, we can't actually prove that this object is a mug. We can prove that the object is stable, and enables drinking based on what we saw with the proof of the cup. The object we're dealing with here, had all the same criteria as our object in the other example in addition. To these thick sides, and being made of clay. So because we had everything else, we were able to easily prove that it's stable and enables drinking. But what we can't do is we can't link protecting its heat to limiting heat transfer. Nothing in my definition of pot actually tells me that the fact that it limits heat transfer actually protects against heat. So, while for my definition of pot I can abstract that because my new object has thick sides and is made of clay, it limits heat transfer, I can't make that last connection to protecting against heat. That is good David. Now let's make sure that we understand the processing that David did. He wanted to show that the object is a mug. So he looked at the conditions, the open conditions of proving that the object is a mug, and there were three of them. For each of them, he tried to build a proof. He could do so for the first two, but this was the open one. So he came up with the closest example, that was the pot. And he did the abstraction here, but he was unable to link these two, because there is no knowledge which links these two at the present time.

11 - Exercise Explanation-Based Learning II

[Click here to watch the video](#)

What knowledge would finish our proof that this object is a mug?

- A Serving Plate
The serving plate protects against heat because it is made of clay and is shaped like an oval.
- An Oven Mitt
The oven mitt is made of cloth. It protects against heat because it limits heat transfer.

- A Wooden Spoon
The spoon is long and made of wood. It protects against heat because it has a handle.
- A Baking Dish
A baking dish protects against heat because it is made of porcelain and has a handle.

Figure 613: Exercise Explanation-Based Learning II

So let us do another exercise that builds on the previous one. Which of this four concepts will enable the agent to complete the proof in the previous exercise?

12 - Exercise Explanation-Based Learning II

[Click here to watch the video](#)

What knowledge would finish our proof that this object is a mug?

○ A Serving Plate

The serving plate protects against heat because it is made of clay and is shaped like an oval.

○ An Oven Mitt

The oven mitt is made of cloth. It protects against heat because it limits heat transfer.

○ A Wooden Spoon

The spoon is long and made of wood. It protects against heat because it has a handle.

○ A Baking Dish

A baking dish protects against heat because it is made of porcelain and has a handle.

Figure 614: Exercise Explanation-Based Learning II

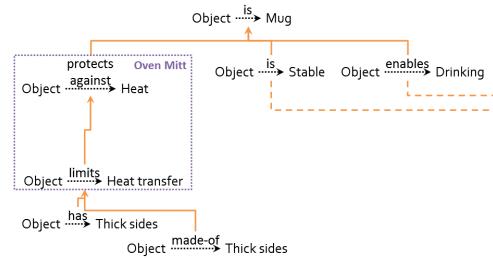


Figure 615: Exercise Explanation-Based Learning II

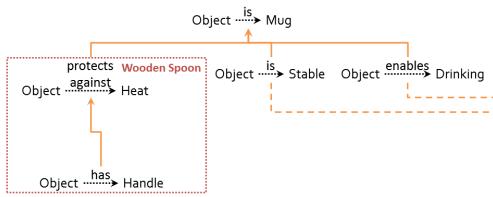


Figure 616: Exercise Explanation-Based Learning II

There are a couple of other points to note about this exercise. An important question in our [Knowledge-Based AI] is, what knowledge does one need. It's not a question of putting

in a lot of knowledge into a program. Instead, the real question is, in order to accomplish a goal what is the minimal amount of knowledge that the AI agent actually needs? Let's see how this applies here. The goal was to show that the object is a mug. Instead of putting in a lot of knowledge, the agent starts asking, what do we need in order to show that the object can protect against heat? What do we need to know to show that the object is stable? And then it goes about searching for that knowledge. The second point to note here is, depending on the background knowledge available, the agent will opportunistically build the right kind of causal proofs. So if the agent knows about the wooden spoon, it will approve this proof. If, on the other hand, the AI agent knew not about the wooden spoon but about the oven mitt, then it could use this particular proof. Which proof the AI agent will build, will depend upon the precise background knowledge available to it

13 - Explanation-Based Learning in the World

[Click here to watch the video](#)

Exploration-based learning is very common in the real world. You and I do it all the time. You need to prop open a door, you bring a chair, and you use it to prop open the door because it just put an explanation for why the chair, in fact, can prop open a door. There is a sheaf of papers on a desk with the shuffling around. You take a coffee mug, put it on the sheaf of paper that acts as a paperweight. Another example or explanation best learned. You and I are constantly dealing with novel situations, we are constantly coming up with creative solutions to them. How do we do it? One way is if we use existing concepts but use them in new ways. We find new connections between them by building explanations for them. This is sometimes called Speed Up learning because we're not learning new concepts, we're simply connecting existing concepts. But it's a very powerful way of dealing with a large number of situations. And today in class, we'll learn how we can build AI agents that can do the same thing that you and I do so well.

14 - Assignment Explanation-Based Learning

[Click here to watch the video](#)

Assignment

How would you use explanation-based learning to design an agent that could answer Raven's progressive matrices?

Figure 617: Assignment Explanation-Based Learning

So how would you use explanation based learning to implement an agent to can solve Raven's progressive matrices? The first question you're asking here is what exactly are you explaining? Are you explaining the answer to the problem, or are you explaining the transformations between figures in the earlier stages of the problem? Given that, what new connections are you learning, Is it learning performed within the problem or old connections an justify the figure to fill in the blank, or to perform the cross problems, where new transmissions and types of problems can be learned and connected together. For example you might imagine that you've encountered two problems before, one are rotation and one are reflection. A new problem might involve both, how do you use those earlier problems to explain the answer to this new problem?

15 - Wrap Up

[Click here to watch the video](#)

To recap...

- Concept space
- Prior knowledge
- Abstraction & transfer

Figure 618: Wrap Up

So today, we've talked about explanation-based learning, a type of learning where we learn

new connections between existing concepts. We first talked about our concept space. A space of information that enables us to draw inferences and connections about existing concepts. We then talked about how prior knowledge is mapped onto this concept space for new reasoning. Then we talked about how we may abstract over prior knowledge to discern transferable nuggets. And how we might then transfer those nuggets onto the new problem we encounter. Next time, we'll expand on this idea of transfer to talk about analogical reasoning, which is inherently transfer-based. Explanation-based learning will also come up significantly in learning about correcting mistakes and in diagnosis. So feel free to jump ahead into those lessons, if you're interested in continuing with this example.

16 - The Cognitive Connection

[Click here to watch the video](#)

Explanation based learning is a very common classroom task, you and I appear to do it all the time. We can use a chair to prop open a door, and we can deliver a very quick explanation for why the chair is a good prop. We can use a coffee mug to hold down a pile of papers, and pull out an explanation for why the coffee mug would make a good paper weight. Of course there is a lot more to it than we have discussed so far. Explanation based learning is central [to Knowledge-Based AI and] on cognitive science, because we are trying to build human like, human level intelligence. Explanations and explanation based learning nowhere is prominent in other schools of AI. Second, note that humans are not very good at explaining everything. We can only explain those things which appear to be consciously accessible. We have a hard time explaining memory processes, for example, or we have a hard time explaining certain kinds of physical actions. For example, when I play tennis, my feet move the way they do, and I can't explain to you why I can't make them move better. Third, although we can generate explanations, this does not necessarily mean that our

process for generating explanation is the same process that we use to arrive at the decision in the first place. Explanations can be post talk. Further, the very act of [to Knowledge-Based AI and] explanations could interfere with the reasoning process. However, when we can generate explanations, it can lead to much deeper, much richer understanding and learning, because it exposes the cause of connections. Finally, for AI systems to be accepted in our society they must be able to generate good explanations. You, for example, will be unlikely to accept the advice of a medical diagnostic system if the diagnostic system cannot explain its answers. I must be able to explain its answers as well as the process it used to arrive at those answers. Explanation is fundamental to trust.

17 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

18 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

Explanation-based learning is type of learning where we learn new connections between existing concepts from the concept space.

References

1. Winston P., Artificial Intelligence, Chapter 8.

Optional Reading:

1. Winston Chapter 17; [Click here](#)

Exercises

None.

Lesson 18 - Analogical Reasoning



Problems are like washing machines. They twist us, spin us and knock us around. But we come out cleaner, brighter and better than before.

– Unknown.

01 - Preview

[Click here to watch the video](#)

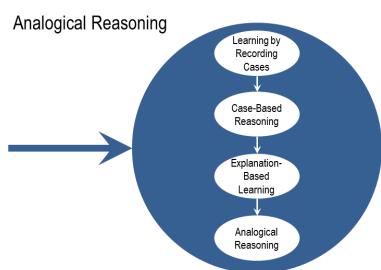


Figure 619: Preview

from known problems across domains. We introduced a notion of transfer previously in explanation based learning. We also have talked about Case Based Reasoning. Today we'll talk about transfer in a much more general manner. We'll start by talking about similarity then revisit case based reasoning. Then we'll talk through the overall process of analogical reasoning, including retrieval, and mapping, and transport. Then we'll close by talking about a specific application of analogy, called Design by Analogy.

02 - Exercise Similarity Ratings

[Click here to watch the video](#)

- Lesson Preview**
- Similarity and case-based reasoning
 - Process of analogical reasoning
 - Design by analogy

A woman is climbing a ladder.

Which of the situations on the right is most similar to the situation above?

Rank them from 1 (most similar) to 7 (least similar)

- A woman climbing a set of stairs.
- An ant walking up the wall.
- A woman painting a ladder.
- A woman climbing the corporate ladder.
- A water bottle sitting on a desk.
- A woman climbing a step ladder.
- A plane taking off into the sky.

Figure 620: Preview

Figure 621: Exercise Similarity Ratings

Today we'll talk about Analogical Reasoning. Analogical Reasoning involves understanding new problems, in terms of family of problems. It also involves addressing new problems, but transferring knowledge of relationships

To illustrate the notion of similarity, let us consider an example. Consider that a woman is climbing up a ladder. Here are seven situations. Can you please rank these seven situations by their order of similarity to the given situation?

03 - Exercise Similarity Ratings

[Click here to watch the video](#)

A woman is climbing a ladder.

Which of the situations on the right is most similar to the situation above?

- 2 A woman climbing a set of stairs.
- 3 An ant walking up the wall.
- 6 A woman painting a ladder.
- 4 A woman climbing the corporate ladder.
- 7 A water bottle sitting on a desk.
- 1 A woman climbing a step ladder.
- 5 A plane taking off into the sky.

Rank them from 1 (most similar) to 7 (least similar)

Figure 622: Exercise Similarity Ratings

Interesting answer, David, note that there are several factors in David's answers. In the two situations that he thought were most similar to a woman climbing up a ladder, in the similarity in the relationship, climbing up as well as similarity between objects, woman and ladder. In contrast, one that he did not think was really similar to a woman climbing up a ladder, woman painting a ladder. Although there is some similarity between the objects woman, and ladder, the relationship is very different here it is climbing up a ladder, here it is painting a ladder which are two very different activities. Between one and two we notice, that both of them have the same relationship, climbing up, but an object is different, In one case it is the step ladder and in another case it is a set of stairs. So one can have similarities in relationships, one can have similarities in objects, of course some of you may have different rankings with the similarities of the one they would give. Because, your background knowledge might be different or your priorities might be different, but the point here is the similarity can be measured around several dimensions. Around the dimension of relationships, around the dimension of objects, around the dimensions of features of objects, and around the dimensions of values of features of objects that are participating in relationships, we'll talk more about this in just a few minutes.

04 - Cases Revisited

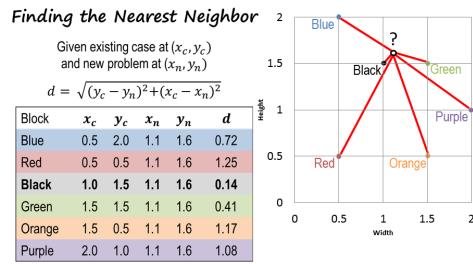
[Click here to watch the video](#)

Figure 623: Cases Revisited

Route	Destination _x	Destination _y
A	10 _E	8 _N
B	1 _E	8 _N
C	10 _E	4 _N
D	2 _E	1 _N
X	8 _E	2 _N
Y	8 _E	2 _N
Z	1 _E	9 _N

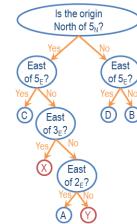


Figure 624: Cases Revisited

We have come across the notion of similarity earlier in this course. When we were discussing learning [by recording] cases, that particular point, we came across the matter of finding the nearest neighbor. At that point we found the nearest neighbor simply by looking at the [by recording] distance between the new situation, and the familiar situations. We came across the notion of similarity when we were discussing case reasoning as well at that point, we came across at least two different methods of organizing the case library. And that, in one method, we could simply organize all the cases in array here's an array of several cases in the domain of navigation and urban area, each case here is represented, by the x and y location of the destination. A different and smarter method, also organizes cases that are discriminatory, the leaf nodes of this discrimination tree represented the cases. The root node and the interior nodes in the discrimination tree represented discrimination, or decisions about the values of specific features for example, east of 5th Street or not east of 5th street, both of these [by recording] schemes are based on measures of similarity. In the first scheme the similarity is based on the similarity between the tags,

If a new problem were to come along it would be more or less similarly one of these cases depending on whether or not its tags match the tags of a particular case here. In the second scheme of this [by recording] tree, similarity is based on, traversing this particular tree, If a new problem came along, we would use the features of that new problem to traverse this tree and find the case whose features best match your new problem. Note that the new problem, and the source cases in all of these examples so far have been in the same domain. Here for example, both the new problem and the source case are in the same domain of navigating in an urban area, in the previous example, the new problem and the source case were the domain of colored blocks in the blocks world. What happens if the new problem and the SOS case are not in the same domain? So consider the example of, a woman walking up the ladder and walking up the wall. The two dimension are the same, we're talking about woman in one case and in other case, a ladder in one case, a wall in other case yeah, there's some similarity. Situations like this, where the new problem and the source case are from different domains, lead to cross-domain analogies. So the question now becomes, how can we find leaf similarity between the new problem, the target problem, and the source case, if they happen to be in different domains?



Figure 626: Need for Cross-Domain Analogy



Figure 627: Need for Cross-Domain Analogy

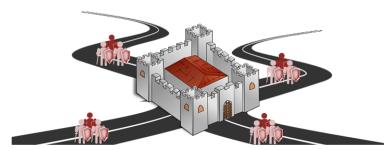


Figure 628: Need for Cross-Domain Analogy

05 - Need for Cross-Domain Analogy

[Click here to watch the video](#)



Figure 625: Need for Cross-Domain Analogy

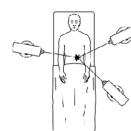


Figure 629: Need for Cross-Domain Analogy

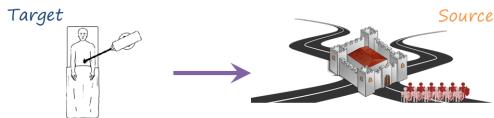


Figure 630: Need for Cross-Domain Analogy

To dig into this issue of similarity between a target problem and a source case in different domains, let us look at another example. Let us suppose it is a patient who has a tumor in his stomach. There is a physician who has a laser gun. She knows that if the laser light were to shine on this tumor, the tumor will be killed and the patient will be cured. But the physician has a problem. The laser light is so strong that it will also kill all the healthy tissue on its way to the tumor, and thereby killing the patient. What should the physician do? This is actually a very famous problem in cognitive science. It was first used by a psychologist called Don Curry around 1926. What do you think the physician should do in this situation? Take a moment and think about it. We'll return to the physician and the patient example in just a minute. First let me tell you another story. Once there was a kingdom ruled by a ruthless king, there was a rebel army approaching the fortress in which the king lived. Well there was a problem. The kings' men had mined, all the roads approaching the fort. As a result, if an army was to walk over the roads, the mines would go off, and the soldiers would be killed. So what did the army decide to do? The army decided to decompose itself into smaller groups, so that each group could come from a different road, and reach the fort at the same time. Because each group was small enough, the mines on the roads did not go off. The soldiers were able to attack the fort at the same time and overthrow the bad king. Now let's go back to the problem of the physician and the patient. What do you think now? Has the answer to the problem changed? Some of you indeed may have changed your answer because of this story

I told you about the king and the rebel army. One solution to this problem is that a physician would divide a very intense laser beam into several smaller, less intense beams. As these beams come from different directions, they do not harm the healthy tissue. However, they reach the tumor at the same time and manage to kill the tumor. You will note that this is an example of cross-domain analogy. Here the target problem had to do with the physician and the patient. The source case had to do with the king and the rebel army. The objects in these two situations were clearly very different. In one case we had a physician and the patient, the laser beam and the tumor. And in the other case we had the king and the rebel army, the fort and the mines. Some of the relationships were very similar. In capturing the fort case, we had a resource, the army, which was decomposed into several small armies, which was sent to the goal location at the same time. We took this battle, we took this strategy, abstracted it out, and then applied it to the patient and physician example. A physician used the same strategy. Resource decomposes into several smaller resources, and sent to the goal at the same time. Now you can also see why the ant climbing a wall is similar to a woman climbing a ladder. The objects are different, ant and wall, woman, and ladder, but the relationship is similar. Climbing up. The cross-domain analogy is then, the objects and the features and the values of the objects can be different. The similarity is based on the relationship. It is the relationship that is important. It is the relationship that gets transferred from the source case to the target problem.

06 - Spectrum of Similarity

[Click here to watch the video](#)

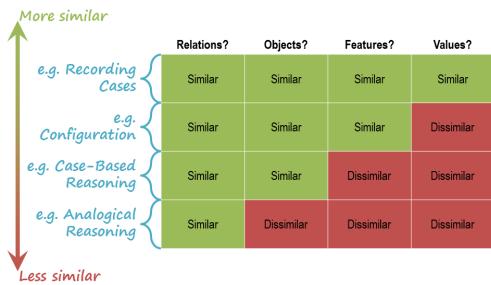


Figure 631: Spectrum of Similarity

We can think of a spectrum of similarity. At one end of the spectrum, the target problem and the source case are identical. At the other extreme end of the similarity spectrum, the target problem and the source case have nothing in common. We can evaluate the similarity between the target problem and source case in similar dimensions. In terms of the relationships occurring in the source case and the target problem. In terms of the objects occurring between the two. In terms of the features of the objects and in terms of the values that the features of the objects take. At the end of the spectrum, where the target problem and the source case are very similar, with relationships, objects, features, and values are all similar. At the other end the values, features and objects may be different, but the relationships are similar. If the relationships too are different, then there'll be nothing in common with the target problem in the source case. When the relationships, objects, features and values are all similar, then that is an example of recording cases, and we have come across it. An example of that was from the colored blocks in the blocks world. When the similarity between the target problem and the source case is along the dimension to relationships and objects, but not along the dimensions of values or values and features, then that's an example of case-based reasoning. We discuss this method in the domain of navigation and urban areas. The objects of the concept between the target problem and the source case being the same means that the domains are the same. So case-based reasoning is within domain analogy. An analogical reasoning in general, objects in the target column and the source case too, might be differ-

ent. We saw an example of analogical reasoning in the Dunker radiation problem, when we were talking about cross-domain analogical transfer. Actually recording cases in case based reasoning are also examples of analogical reasoning, except that they occur in the same domain. The target firm and the source cases in the same domain, which is why we consider them earlier. By analogical reasoning here, we mean cross domain analogical transfer. As in the Dunker radiation problem.

07 - Process of Analogical Reasoning

[Click here to watch the video](#)

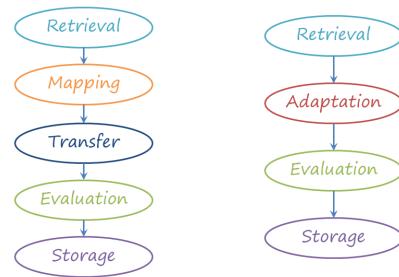


Figure 632: Process of Analogical Reasoning

Analogical reasoning allows us to look at new problems in terms of familiar problems. It also allows us to transfer knowledge from familiar problems to new problems. A hierarchy process of analogical reasoning is shown here. It consists of five major phases, retrieval, mapping, transfer, evaluation, and storage. We'll discuss all five stages, in detail. Let us compare for a moment, the process for an illogical reasoning in general, with a process for, case based reasoning, within domain and illogical reasoning that we discussed earlier. Notice that retrieval, evaluation, and storage, are common between the two processes. In case based reasoning the target problem and this first case, were from the same domain. They had the same kind of relationships, and the same kind of objects. We simply had to adapt the source case to address the target properly. An analogical reasoning in general, the target form and the source case need not be from the same domain. When they are not from the same domain, we can't just take

the source codes and adapt it. We first have to map the target problem with the source case that is, we need to address the correspondence problem. What in the target problem, corresponds to what in this source case as an example? The laser beam and the target Duncker's radiation problem corresponds to the rebel army in the source case. Once we have mapped the conceptual relationship in the target problem to the conceptual relationships in the source case, then, we can try to transfer some of the relationships in the source case to the target problem. We can first abstract those relationships and then transfer them to the target problem. As an example, the Duncker's radiation problem we first did the alignment, that is, we just did the correspondence problem, what in the target [Euclidean] corresponds to what in the source case? Then we took the relationship, and abstracted it. The relationship in that case was, take the resource and decompose it into several smaller resources, and send them to the goal at the same location. That particular relationship, that particular pattern, is what we abstracted and transferred to the target problem. Note that this is just one theory of analogical reasoning. In other theories, some of these boxes are configured differently. For example, in another theory, mapping is a part of retrieval. We do mapping in order to do retrieval.

08 - Analogical Retrieval

[Click here to watch the video](#)

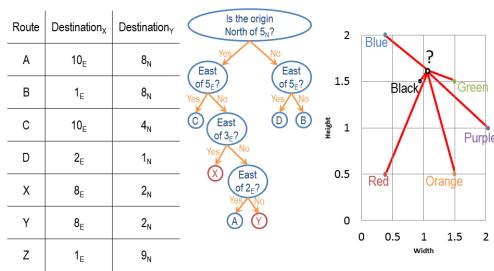


Figure 633: Analogical Retrieval

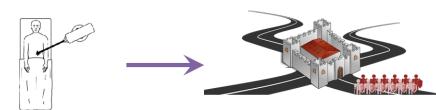


Figure 634: Analogical Retrieval

- | | |
|---|---|
| Superficial Similarity <ul style="list-style-type: none"> • Features • Counts • Objects | Deep Similarity <ul style="list-style-type: none"> • Relationships between objects • Relationships between relationships |
|---|---|



Figure 635: Analogical Retrieval

Let us look at analogical retrieval more closely. Once again, we have come across this idea earlier. We did analogical retrieval in recording cases by the k nearest neighbor method. We did analogical retrieval on a case based reasoning using two different methods, the array method and the discrimination tree method. Here the criteria for evaluating similarity were very clear, as we discussed earlier. [case indexing] distance, same tags, as well as placement in this discrimination tree. The question now becomes, what criteria should be used to decide on the similarity between the target problem and the source case, given that they come from different domains? On surface, there seems to be little similar between the two. None of the objects are similar. None of the values of features are similar. Yet, there is a deep similarity there. We can distinguish now between superficial similarity between two situations and deep similarity between two situations. Superficial similarity deals with features of objects or counts of objects or objects themselves. Deep similarity deals with relationships between objects, or sometimes relationships between relationships. Examples of this arise from the variables test with which you

are already familiar. Features here refer to the size of the square, the size of the circle, or perhaps where there is a hollow square, or a solid dot. The count refers to the number of squares, or number of circles, in a particular image. Objects here refer to circles and squares and dots. Let us look at relationship between objects. Two situations are said to be deeply similar, if the relationship between the objects is similar. As an example, a and b are similar, in that, that the dot is outside the circle here and the square is outside the circle here. A and b are also similar in that the dot is above the circle here and the square is above the circle here. What about relationships between relationships? Let us compare a and b. In going from A to B, the dot has disappeared, and a square has come outside the circle, and become bigger. Now we can compare this relationship between a and b, with some of the relationship between a c and a b, in which too, some object might be disappearing, and another object which may be in the center of the circle comes out of the circle. I'm sure you've come across problems like that on the variable test. This is an example of a binary relationship, a relationship between two objects. This is an example of a higher order relationship, a tertiary relationship if you wish. This is a relationship between the relationship between objects. You might even say that these are examples of unary relationships. These are just examples of objects and their features and counts. In general, as we go from unary relationships to binary relationships to tertiary relationships to even higher order relationships, the similarity becomes deeper and deeper. This means that mind decides two situations to be more similar if the similarity is at the level of relationship between objects rather simply at the level of objects or features or counts and objects.

<i>Types of Similarity</i>		
Semantic Conceptual similarity between the target problem and the source case.	Pragmatic Similarity of external factors, such as goals.	Structural Similarity between representational structures.

Figure 636: Three Types of Similarity

Semantic similarity used with conceptual similarity between the target problem and the source case. If we recall the original exercise that David had answered, in that exercise, a woman climbing up a ladder is conceptually similar, semantically similar to a woman climbing up a step ladder. The same kind of concepts occur in both situations. Woman, and step ladder or ladder. Pragmatic similarity concerns with external factors. Factors external to the presentation, such as goals. As an example, in the Dunker radiation problem, the physician had a goal of killing the tumor, which was similar to the goal of capturing the fort in case of the rebel army and the king. Pragmatic similarity refers to similarity of external factors, factors external to the representation, such as similarity of goals. The Dunker radiation problem for example, the physician had the goal of killing the tumor, which was similar to the goal of capturing the fort in case of the rebel army in the king example. The third measure of similarity is structural similarity. Structure here refers to the structure of presentations, not to physical structure. Now structural similarity of the first two, similarity between the representational structures of the target problem and the source case, and we'll look at an example of this in just a few minutes. Know that one can assign different kinds of weights to these three measures of similarity. So some queries of analogy focus on structural similarity. Other theories of analogy focus on semantic and pragmatic similarity. That is also why you may have given slightly different answers to the questions in the first exercise than David did.

09 - Three Types of Similarity
[Click here to watch the video](#)

10 - Exercise Analogical Retrieval I
[Click here to watch the video](#)

A woman is climbing a ladder.

Mark whether each situation has deep similarity, superficial similarity, both, or neither with the situation above.

Deep	Superficial
<input type="radio"/>	<input type="radio"/>

A woman climbing a set of stairs.
An ant walking up the wall.
A woman painting a ladder.
A woman climbing the corporate ladder.
A water bottle sitting on a desk.
A woman climbing a step ladder.
A plane taking off into the sky.

Figure 637: Exercise Analogical Retrieval I

Let us do another exercise together now that we know about deep similarity and superficial similarity. Consider the situation again, a woman is climbing up a ladder. Give this set of situations, mark whether each of the situations is deeply similar or superficially similar to this given situation. Know that some might be both and others might be neither

11 - Exercise Analogical Retrieval I

[Click here to watch the video](#)

A woman is climbing a ladder.

Mark whether each situation has deep similarity, superficial similarity, both, or neither with the situation above.

Deep	Superficial
<input checked="" type="radio"/>	<input type="radio"/>
<input checked="" type="radio"/>	<input type="radio"/>
<input type="radio"/>	<input checked="" type="radio"/>
<input checked="" type="radio"/>	<input type="radio"/>
<input type="radio"/>	<input type="radio"/>
<input checked="" type="radio"/>	<input type="radio"/>
<input type="radio"/>	<input type="radio"/>
<input checked="" type="radio"/>	<input type="radio"/>
<input type="radio"/>	<input type="radio"/>

A woman climbing a set of stairs.
An ant walking up the wall.
A woman painting a ladder.
A woman climbing the corporate ladder.
A water bottle sitting on a desk.
A woman climbing a step ladder.
A plane taking off into the sky.

Figure 638: Exercise Analogical Retrieval I

What do you think, David? So perhaps unsurprisingly, I said that the two most similar situations in my original answer were both deeply and superficially similar. A woman climbing a step ladder is both deeply similar in that the relationship are the same. And superficially similar in that the woman and the ladder are somewhat similar. I also said that the woman climbing a set of stairs is both deep and superficially similar. We still have a woman climbing in both, and I said that a set of stairs is somewhat superficially similar to a ladder, in that both are designed to allow someone to ascend. Although some others might say that that's not superficially similar. Then I said that the other three

that involved rising in some way were all deeply similar as well. The plane taking off into the sky, the woman climbing the corporate ladder, and the ant walking up the wall were all deeply similar to me because all involved this relationship of climbing. I said that the woman painting a ladder was only superficially similar, which is why it was one of the lowest rated in our original exercise. And finally the water bottle sitting on a desk is neither deeply similar nor superficially similar, because nothing is the same between those two situations. This is good, David. Once again, different people may give different answers to this exercise. Why did we do so? Well, let's examine it next.

12 - Exercise Analogical Retrieval II

[Click here to watch the video](#)

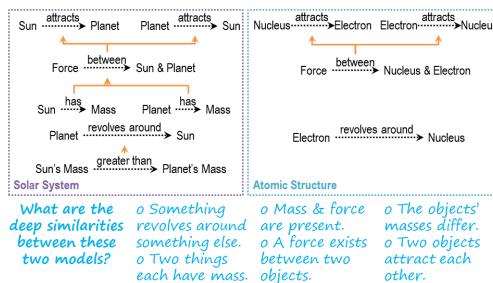


Figure 639: Exercise Analogical Retrieval II

Many science textbooks in middle school or high school explain the atomic structure in terms of the solar system. Here's a representation for the solar system, here's a representation for the atomic structure. Let us see how this model of the solar system helps us make sense of the atomic structure. We'll use this example often going forward. And this representation of the solar system is arrows are denoting causality. So the sun's mass is greater than the planet's mass, which causes the planet to revolve around the sun. Similarly for the atomic structure, there is a force within the nucleus and the electron, and that causes the nucleus to attract the electron and the electron to attract the nucleus. Given these two models, what are the deep similarities between them?

13 - Exercise Analogical Retrieval II

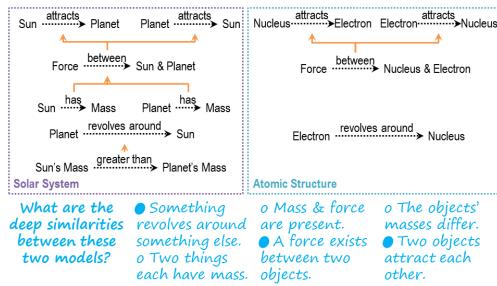
[Click here to watch the video](#)

Figure 640: Exercise Analogical Retrieval II

What do you think, David? What are the deep similarities here? So, I picked that three of these are deep similarities. In both there's something revolving around something else. The planet revolving around the sun and the electron revolving around the nucleus. In both, there's a force between two objects. A force between the sun and the planet, and a force between a nucleus and an electron. And in both, two objects attract each other. All three of these are based on relationships. The revolution, the force between two objects and the attraction between two objects. I didn't pick the other three because while these two are true, they don't represent deep relationships. And while this is actually true, it's not actually captured in our representation. So this isn't part of our knowledge base for the purpose of this problem. Now we can see why the textbooks write about this solar system and the atomic structure in such a way that these relationships become salient. They help us make sense of the atomic structure, but pointing to the deep similarities between the relationship that are occurring with the atomic structure and the relationship that are occurring in the solar system



Figure 641: Analogical Mapping

Now let us consider analogical mapping. The problem here is called the correspondent's problem. There are a number of obvious relationships in this target problem. There are a number of obvious relationships in this source case. What in the target problem corresponds to what in the source case? If we can address the correspondence problem. If we can say, for example, that the laser beam corresponds to the rebel army, then we can start aligning the target problem and the source case so it makes the deep similarities between relationships salient. Note there are several parts of a target problem and several in the source case. In principle, any of these objects of the target problem could correspond to any of the objects in the source case. In which case we would have an m to n mapping, and that becomes computationally inefficient. If you and I, often do not have much of a problem deciding, if the laser beam must correspond to the devil army. How do we do it? And how can we help AI agents make similar kind of correspondences? Our answer is, we'll make use of relationships. In fact, we'll make use of higher order relationships, whenever possible. We'll give precedence to higher order relationships, over other relationships. As a unary relationship, we might say that a patient is a person here, and king is a person there. The binary relationship we might say, that physician has a resource, the laser beam. And that the rebel army has a resource, the army itself. It's a higher ordered relationship, a tertiary relationship between say, that between the goal and the resource is an obstacle. They held a tissue in this case. Similarly between the goal and the resource is an obstacle, the minds in this case. We focus on the higher ordered relation-

14 - Analogical Mapping

[Click here to watch the video](#)

ship there, that's where the deepest similarity between the two situations lies. This is how we know to mark between the king and the tumor and not between the king and the patient. Although the king and the patient are superficially similar, a deeper similarity lies in viewing the king and the tumor in terms of goals which need to be cured or captured using a resource when there is an obstacle in between them.

15 - Exercise Analogical Mapping

[Click here to watch the video](#)

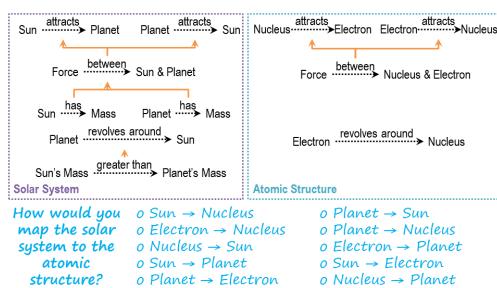


Figure 642: Exercise Analogical Mapping

Let us do an exercise on deep relationships. Let's get back to for example the solar system and the atomic structure. Let us suppose that you've given this representation of the solar system and this representation of the atomic structure. How would you map the solar system to the atomic structure?

16 - Exercise Analogical Mapping

[Click here to watch the video](#)

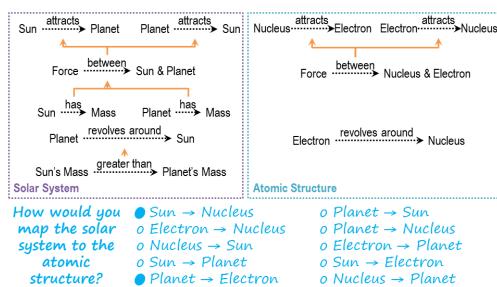


Figure 643: Exercise Analogical Mapping

What choices did you pick David? So there's a few different ways we can do the mapping between the sun, the planet on the left and the

nucleus and the electron on the right. If we look only at the top half of each representation all we can really say is that the sun must map to one of them and the planet must map to one of them. Because these relationships are symmetrical, if I don't consider the bottom half it wouldn't matter which is which. If I look at the bottom half of the representations however, it tells me which part of the Solar System must correspond to which part of the Atomic Structure. The planet revolves around the Sun, and the electron revolves around the nucleus. So the planet and the electron must correspond because both are doing the revolving and the Sun and the nucleus must correspond because they are the centers of the revolution. So, the sun corresponds to the nucleus and the planet corresponds to the electron. This is right, David. Another thing to take away from here is, look at the depth of understanding it requires in order to be able to make the right kind of correspondences. If one did not have the right kind of model for the solar system and atomic structure that captures the deep relationships, then the mapping may not be done and the alignment wouldn't work. And we would not be able to understand the atomic structure in terms of the solar system. This models deep and rich models of the two systems, the target problem and the source case are essential to deciding how to align them, how to map them, and as we will see in a moment, what to transfer and how to transfer it.

17 - Analogical Transfer

[Click here to watch the video](#)

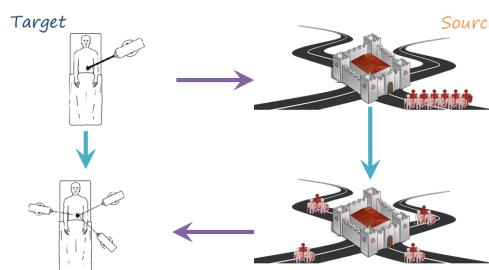


Figure 644: Analogical Transfer

Now let us consider analogical transfer. So, given this target problem, analogical retrieval

has led to this source case. Given a model of the target problem and a model of the source case, analogical mapping has also occurred, correspondence has been established. This we now know that a king corresponds to the tumor, not to the patient. And that the rebel army corresponds to laser beam. For the source case we also know the solution. The rebel army divided itself into smaller groups, and the smaller groups all arrived at the fort at the same time. Now the question becomes how can we transfer this solution to our original target problem. From the source case now, we're inducing a pattern of relationships, a strategy. In this case the pattern is that if there is a goal, capturing the king, and a resource, the rebel army and an obstacle between the resource and the goal. They march on the road, then decompose the resource into several smaller resources. And send them to the goal from different directions at the same time. This abstract pattern is now transferred to the target problem and instantiated. Because we know that the goal is the tumor, the resource is the laser gun, and the obstacle is the healthy tissue, we know what to do. We must decompose the resource, the laser gun, into smaller pieces, smaller, less intense laser beams, and send them to the tumor, the goal, at the same time from different directions. This is how we can transfer the problem solving strategy, from the source case to the target problem. Note that this transfer depended upon the correct mapping, the correct alignment between the target problem and the source case, which in turn depended upon the retrieval of the source case corresponding to this target problem. Note the important rule or goal here. The goal was to capture the king. So this is an example of pragmatic similarity. With a lot of similarities at the level of the goal, capturing the king, curing the tumor.

18 - Exercise Analogical Transfer
[Click here to watch the video](#)

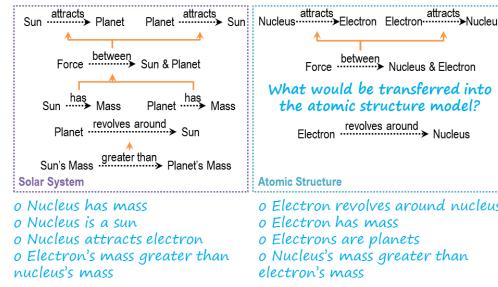


Figure 645: Exercise Analogical Transfer

Let us do an exercise on analogical transfer together. Back to our example of this sort of system in the atomic structure. Given this representation of the solar system and this representation of the atomic structure, what would be transferred from the solar system into the atomic structure model?

19 - Exercise Analogical Transfer

[Click here to watch the video](#)

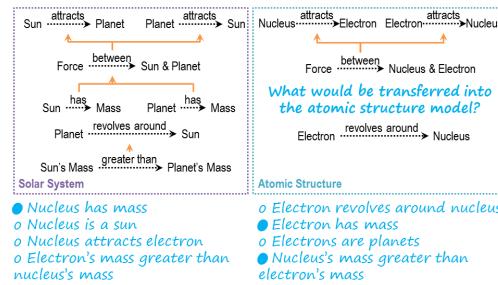


Figure 646: Exercise Analogical Transfer

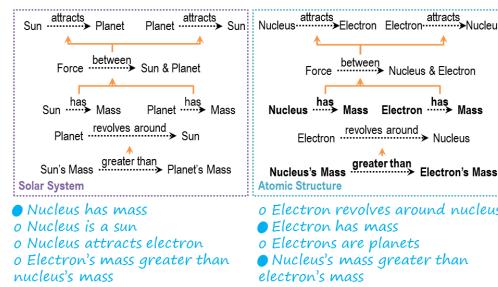


Figure 647: Exercise Analogical Transfer

That's a smart answer David. Recall that originally I had said that I'll explain structural similarity, and I have not done it so far. I'm going to use the spherical example and David's

answer to explain it now. Given the solar system as the source case, and the atomic structure as target problem. We can see that there is little semantic similarity between them. The kinds of objects that occur in the solar system are not at all like the kinds of objects that occur in the atomic structure. We can also see that pragmatic similarity is not a major issue here. We're not talking about the goal of the solar system or the goal of the atomic structure. Although we might have the goal of understanding atomic structure in the solar system, there is nothing in the solar system, or in the atomic structure, which has a goal. Yet, David was able to answer this question correctly. This is because of structural similarity. Let us consider the top part of the model of the solar system. You can think of this top part like a graph. The vertices in this graph, correspond to objects and their properties. The edges in this graph correspond to relationships, such as force between Sun and Planet. Or, Sun attracts Planet, and Planet attracts Sun. Once again, this graphical representation of the model atomic structure. The words are representing the objects and the features. And the edges are representing the relationships between the objects. Although there's little semantic similarity, or pragmatic similarity between the two situations, we can see a structural similarity. A similarity in the structure of the graphs. Because this part of the graph of the representative of the solar system. The similar this part of the graph of the representative of the atomic structure. We can differ infer that we can transfer this part of the graph of the solar system to infer this part of the graph of the atomic structure. Structural similarity then captures relational similarity. What is common between these two situations is neither the objects or the goals. Where as common here as the relational similarity, and that is what structure similarity captures.

20 - Evaluation and Storage in Analogical Reasoning
[Click here to watch the video](#)

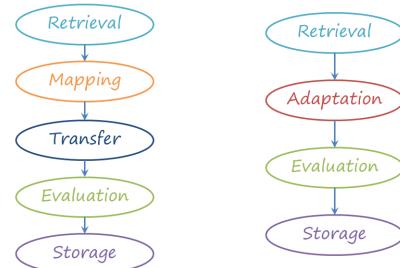


Figure 648: Evaluation and Storage in Analogical Reasoning

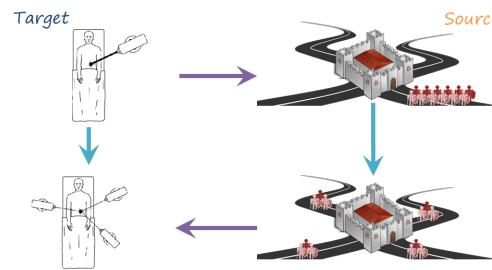


Figure 649: Evaluation and Storage in Analogical Reasoning

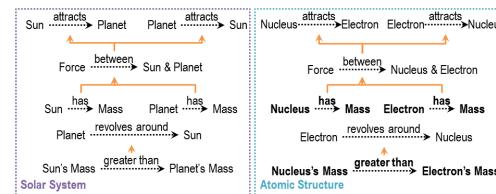


Figure 650: Evaluation and Storage in Analogical Reasoning

Let us briefly talk about evaluation in storage. These evaluation and storage steps in analogical reasoning are very similar to the evaluation and storage steps in case based reasoning. Analogical reasoning by itself does not provide guarantees of correctness. So the solution that it proposes must be evaluated by some manner. For the down correlation problem, for example, we may evaluate the proposed solution by doing a simulation. Once the evaluation has been done, then the new problem and a solution can be encapsulated as a new case and stored back.

in memory for later potential reuse. To return to the down correlation problem, as an example. Once we have the solution of decomposing the laser beam into several smaller beams and sending them to a tumor at the same time from different directions, we can do a simulation of this solution and see whether they are successful. If it is, then we can encapsulate the target problem and the proposed solution as a case, and store it in memory. It might be useful later. It could potentially become a source case for a new target problem to come later. Once again, in this way, the AI agent learns incrementally. Each time it solves a problem, the new problem and its solution becomes a case for later reuse.

21 - Design by Analogy

[Click here to watch the video](#)



Figure 651: Design by Analogy

It is often useful to look at specific problem domains, both to see how we can apply current theories of chronological reasoning to them and also to see how we can use those problem domains to build new theories of chronological reasoning. So let us turn now to the domain of design. In design, there is a new movement that is sometimes called biologically inspired design, or biomimicry. This movement is pulled by the need for environmental sustainability and is pushed by the desire for creativity and innovation and design. On the top left here is a picture of the Shinkansen 500 train in Japan. This is a bullet train. It's called a bullet train because of the shape of its nose. This particular shape is inspired by the shape of the beak of the Kingfisher. The story goes something like this. Japanese railway engineers were interested in

building faster trains. However, they had a problem, these trains had to go through tunnels. And as they went through tunnels, they created shock waves, which created a lot of noise, bothering the neighbors. The shock wave was created because outside the tunnel and inside the tunnel were two different mediums. By serendipity, the railway engineers looked at how the Kingfisher goes from the medium of air into a medium of water, dips its beak and catches its prey. The shape of its nose allows us to create a smaller shock wave. We use the same principle to create the design of the nose of the bullet train. Shinkansen 500 travels faster than previous trains and also makes less noise than previous trains, mostly because of the nose shape. Another example often cited in biomimicry is the example of a Mercedes Benz box car, designed by inspiration to the Box-fish. Notice as biological inspired design entails analogical reasoning. There's a target problem. There's a source case. There is cross-domain analogical transfer.

22 - Design by Analogy Retrieval

[Click here to watch the video](#)



Figure 652: Design by Analogy Retrieval

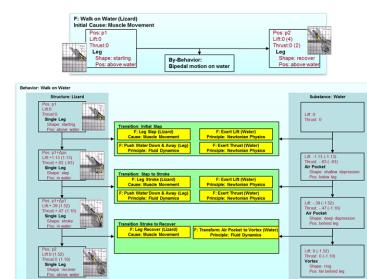


Figure 653: Design by Analogy Retrieval

To illustrate analogical reasoning and design, or analogical design, we'll talk about a specific problem, let us suppose we design a robot that can walk on water. Nature already offers several examples of organisms that can walk on water, this is a picture of the basilisk lizard, which can walk very well on water and catch its prey. Recall that we said earlier that for analogical mapping and crossword worker row, requires a deep understanding of the source case and the target problem. That is true here as well in case of analogical design, we require a deep understanding of the locomotion of the basilisk lizard, in order to be able to design a robot that can walk on water, inspired by the design of the basilisk lizard. Here is a model of the basilisk lizard, this model is sometimes called structure behavior function model. This particular picture doesn't show the structure, it just shows the function and the behavior. The function is shown at the top here, It is shown by its initial state and its goal state, and its function is achieved by behavior shown here. The behavior is represented as a series of states, and transitions between those states. We will not talk about the representations in more detail here, readings given in the class notes give this sort of representations a lot more detail if you are so interested.

23 - Design by Analogy Mapping Transfer

[Click here to watch the video](#)

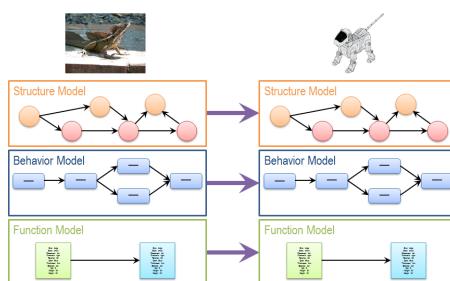


Figure 654: Design by Analogy Mapping Transfer



Figure 655: Design by Analogy Mapping Transfer

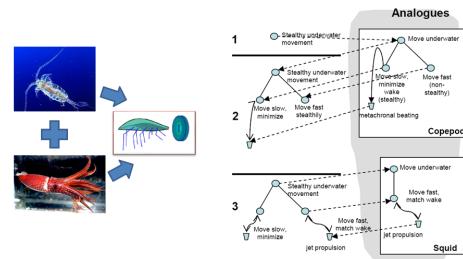


Figure 656: Design by Analogy Mapping Transfer



Figure 657: Design by Analogy Mapping Transfer

Recall that we started with a problem of designing a robot, that can walk on water. Let us suppose that, that particular target problem resolves in the retrieval of a source case, of a robot design that we already encountered. One that can walk on ground. Now the question becomes, how can we adapt this particular design of the robot that can walk on the ground, into a robot design that can walk on water? Let us now suppose, if we reuse this particular problem of designing a robot to walk on water. As a probe into the case memory. And now the case returns

to us, the design of the basilisk lizard. That might happen, because the design of the basilisk lizard, is indexed by its functional model, walk on water. So there is a pragmatic similarity between the two. We now have the design of a robot who can walk on ground, and we have the design of a biological organism, the Basilisk Lizard, that can walk on water. For the Basilisk lizard, we also have a complete model, a complete explanation of how its structure achieves its function. Now that we have a partial design for the robot, this is a design of the robot that can walk on ground. And we have a design of an organism that can walk on water. We can try to do an alignment between these two. This alignment will be based on the similarity between relationships. Clearly, the objects here, and objects there are very different. Once we have aligned these structural models, or the robot that can walk on ground, and the basilisk lizard that can walk on water. Then, we can start doing transfer. We can transfer specific features, of the structure of the basilisk lizard. For example, the shape of its feet, to this model, of the robot that can walk on ground. In order to convert it into a robot, it can walk on water. Having constructed a structural model, for this robot that can walk on water then we can try to transport the behavioral model, and then the functional model. And then this way we have a complete model of a robot that can walk on water. Along with an explanation of how it will achieve its function. This is sometimes called compositional analogy. We'll first do mapping at the level of structure, and that mapping at a level of structure helps us transfer some information. That in turn allows us to transfer information at the behavioral level. Once we have transferred information at the behavioral level, we can climb up this abstraction hierarchy, and transfer information at a functional level. We can now revisit our computational process, and our logical reasoning. Initially we had presented this particular process like, a linear chain, Retrieval, Mapping, Transfer, Evaluation and Storage. In general, however, there can be many loops here. We may do some initial mapping, for example, that may result in some transfer of information.

But that transfer then, may lead to additional mapping, and then to additional transfer and so on. Here is another brief example, from biological inspired design, in this case we want to design a robot that can swim under water in a very slowly manner. This particular function of swimming underwater in a stealthy manner, reminds a design team of a copepod. A copepod is a biological organism, that has a large number of appendages. It moves underwater, in such a way that it generates minimum wake, especially when it moves very slowly. On the other hand, when it moves rapidly, then the wake becomes large, when the wake is small then its motion is very steady, when the wake is large, its motion is no longer steady. An analogically transfer of knowledge about this particular copepod, gives a design for the microbot for slow velocity. This analogy, decomposes our original design problem. We had the original design problem, as moving underwater in a stealthy manner. Now that we have a design of an organism, for moving underwater at low velocities, we are still left with the sub goal of moving underwater at high velocities. The goal of designing a microbot, that can move underwater in a stealthy manner, at fast velocities, may remind the design team of the squid. The squid uses a special mechanism, like the jet propulsion mechanism to move underwater in a stealthy manner at pretty high velocities. Now we have created a designed for microbot. Where part of the design comes from the design of the copepod, and the other part comes from the design of the squid. Instead of borrowing the design from one source case, we are borrowing parts of the design of multiple source cases. This is a compound analogy. Notice that there's a problem evolution going on, which started with one problem. We arrived at a partial solution to that. Which then leads us to a problem evolution. And the problem transformation. We then have a new understanding of the problem. So, this example we saw, how we first did analogical retrieval of the coco powder organism. Then Mapping, then Transfer. That then lead to addition retrieval, in this case with a squid. Once again this process is not linear. Just like we can iterate between Mapping and Transfer, similarly

we can iterate between Transfer and Retrieval.

24 - Design by Analogy Evaluation Storage

[Click here to watch the video](#)



Figure 658: Design by Analogy Evaluation Storage

Evaluation too can play a very important role in the iterative loops in this analogical reasoning process. One can use several different methods for doing evaluation. In the robot that can walk on water, for example, we can do a simulation, or we can build a physical prototype. If the evolution succeeds, then well and good, we can encapsulate the target polymer solution as a case and store it in case memory. If the evaluation fails, we may need to revisit transfer and see whether we want to transfer some of the knowledge or revisit mapping, and perhaps align things differently or revisit retrieval and perhaps try to retrieve a different source case. As an example, supposing that the evaluation shows, then the robot that we designed for walking on water is a little too heavy. In that particular case, we may change the problem specification and retrieve a different kind of organism that perhaps is a little lighter. Let us suppose that we evaluate the design of the robot that can walk on water and find that the design is a little too heavy. In that case, we can go back to the transfer stage and see whether we can transfer some of the relationship that might make the robot a little lighter. Or we can go back to the mapping stage and see whether we can align the source case and the target problem slightly differently so that we can transfer a different relationship. Or alternatively, we can go back to the retrieval state and try to retrieve a source case, a different kind

of biological organism altogether. Thus, we see that the process of analogical reasoning is not linear at all and see it can have many different kinds of iterations. Analogical reasoning continues to be an important topic in our research and biological-inspired design is becoming one. We provide several readings with both topics in the class notes.

25 - Advanced Open Issues in Analogy

[Click here to watch the video](#)

Advanced questions in analogical reasoning

- Common vocabulary
- Abstraction and transformation
- Compound and compositional analogies
- Visuospatial analogies
- Conceptual combination

Figure 659: Advanced Open Issues in Analogy

There're a number of advanced and open issues in analogical reasoning, that are the subject for current research. First, because analogical reasoning entails cross-domain transfer, does it mean that we necessarily need a common vocabulary across all the domains? Consider the example of the atomic structure and the solar system once again. Suppose I were to use this term revolve, to say the electron revolves around the nucleus. But use the term rotate to say the planet rotates in an orbit around the sun. I have used two different terms. How then can I do alignment between these two situations? Should I use the same vocabulary? If I don't use the same vocabulary, what alternative is there? Second, analogical reasoning entails problem abstraction and transformation. So far we have talked as if the problem remain fixed, it's source case is retrieved and transferred across. But often, the agent needs to abstract and transfer the problem, in order to be able to retrieve the source case. A third issue in analogical reasoning concerns compound and compositional analogies. So far we have talked that given a problem, we can retrieve a case and transfer some knowledge from

that case to the problem. But often we retrieve not one case, and we transfer knowledge from not one case, but from several cases. If you're designing a car, you might design the engine binology to one vehicle and the chassis binology to some other vehicle. This is an example of compound analogy. But how can we make compound analogy work? In compositional analogy, analogy works at several levels of abstraction. Supposing we were to make an analogy between your business organisation and some other business organisation. We might make this compositional analogy, first at the level of people. Next to the level of processes. Third of level of the organisation as a whole. This is another example of compositional analogy, where mapping at one level supports transfer to the next level. How do we do compositional analogy in AI agents? Fourth, visuospatial analogies. So far we have talked about analogies in which it transferred necessarily engages causal knowledge. But a large number of analogies in which causality is at most implicit. We'll consider these visuospatial analogies later in the class. Fifth, conceptual combination. A powerful learning mechanism is learning a new concept by combining parts of familiar concepts. Analogical reasoning is one mechanism for conceptual combination. I have a one concept, [discrimination] concept, that of the atomic structure, another concept, the solution concept. The concept of the solar system. I take some part of the solar system knowledge, combine it with my concept of the atom to get a new concept of the atom. If you're interested in any of these issues, I invite you to join the PhD program in Computer Science.

26 - Assignment Analogical Reasoning
[Click here to watch the video](#)

Assignment

How would you use analogical reasoning to design an agent that could answer Raven's progressive matrices?

Figure 660: Assignment Analogical Reasoning

So how would you use analogical reasoning to design an agent to answer Raven's progressive matrices? This might be a tough question at first, because the agents we're designing only operate in one domain, taking the Raven's test. They don't look at other areas. So, we're going to get the knowledge necessary to do cross domain analogical transfer. In this instance instead of the agent doing the analogical reasoning, maybe it's you doing the analogical reasoning. Can you take inspiration from other activities to inspire how you design your agent? Or can you take knowledge from other activities and put them in your agent, so that it can do the analogical reasoning?

27 - Wrap Up

[Click here to watch the video](#)

To recap...

- Similarity
- Analogical retrieval
- Analogical mapping
- Analogical transfer
- Evaluation and storage
- Design by analogy

Figure 661: Wrap Up

So today, we've been talking about analogical reasoning. We started by talking about similarity. As we saw in our opening exercise, similarity is something that we evaluate very easily without even really thinking about it. How can we design agents that can do the same kind of similarity evaluation? We then talked about

analogical retrieval, which can be difficult, because we're trying to retrieve examples across other domains. How can we structure our knowledge to facilitate this kind of retrieval? How can a system know the given a model of the atom, it should retrieve a model of the solar system? Then we talked about mapping, which is figuring out which parts of different systems correspond. For example, how can figure out that the troops in the four example correspond to the lasers in the tumor example? We then talk about transfer, which is moving knowledge from the concept we know to the concept we don't. For example, we used what we knew about the solar system to fill in our knowledge of the atom. Then next, we talked about evaluation and storage. How do we evaluate our analogies? In the tumor example, we might actually try that medical procedure. But for other analogies, how do we evaluate them? And then how do we store them for future use? Last, we talked about a special kind of analogy, design by analogy, where we use something that we know a lot about to inform our design of something new. We'll talk a lot more about this, especially design by analogy, when we come to the design unit later in our course.

28 - The Cognitive Connection

[Click here to watch the video](#)

Analogy is often considered to be a core process of cognition. A common example of analogy we encounter everyday is that of metaphors. For example, you can imagine someone saying, I had to break up with her. We had grown very far apart. Far apart here is a spatial metaphor. One of the famous examples of metaphors comes from Shakespeare. All the world's a stage, all the men and women merely players. The theater here is a metaphor for the world. A third connection is the Rubin's test of intelligence. The Rubin's test is considered to be one of the most common

and reliable test of intelligence, and as you well know by now, it is based entirely on analogies. An analogy is that central to cognition.

29 - Final Quiz

[Click here to watch the video](#)

Please summarize, what you learned, in this lesson.

30 - Final Quiz

[Click here to watch the video](#)

Great, thank you for your answer.

Summary

Analogical-reasoning is inherently transfer-based learning where we abstract over prior knowledge to discern tranferable nuggets of knowledge and then transfer them onto a new problem in order to solve the new problem.

References

1. Goel Ashok, Bhatta Sambasiva, Use of design patterns in analogy-based design.
2. Falkenhainer Brian, Forbus Kenneth, The Structure-Mapping Engine: Algorithm and Examples.

Optional Reading:

1. The Structure-Mapping Engine: Algorithm and Examples; T-Square Resources (Analogical Reasoning 1.pdf)
2. Use of design patterns in analogy-based design; T-Square Resources (Analogical Reasoning 2.pdf)

Exercises

None.

Lesson 19 - Version Spaces



Natural Selection is the blind watchmaker, blind because it does not see ahead, does not plan consequences, has no purpose in view. Yet the living results of natural selection overwhelmingly impress us with the appearance of design as if by a master watchmaker, impress us with the illusion of design and planning.

— Richard Dawkins, *The Blind Watchmaker*.

01 - Preview

[Click here to watch the video](#)

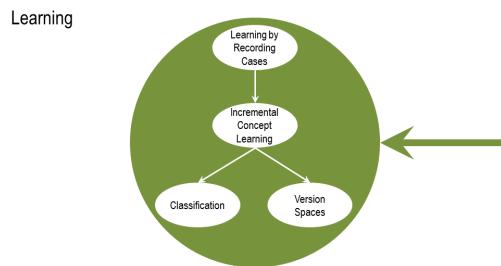


Figure 662: Preview

closely connected to the issue of incremental concept learning. In both cases, there are small sort of examples arriving one at a time. In incremental concept learning, we made use of background knowledge. In generalization and learning, background knowledge may not be available. Similarly, in incremental concept learning, we control the order in which the examples arrived, but in general, in generalization in learning, we may have no control over the ordering of examples. Today, we'll start by defining version spaces, we'll talk about version spaces that has an abstract technique. Then we'll go into the algorithm for executing the technique of version spaces. We'll also, towards the end, talk about an alternative method for organizing the examples into a decision tree, or an identification tree. This method is very similar to the method of incremental discrimination tree learning that we learned when we were talking about case based reasoning

Lesson Preview

- Definition
- Abstract version spaces
- Algorithm for version spaces
- Identification trees

Figure 663: Preview

Today, we'll talk about generalization and learning, using the method of version spaces. The issue of generalization and learning, is

02 - Incremental Concept Learning Revisited

[Click here to watch the video](#)

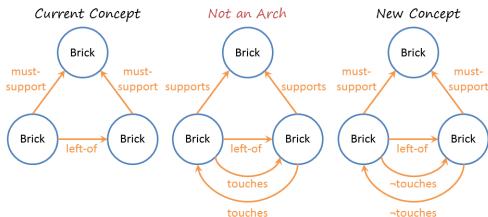


Figure 664: Incremental Concept Learning Revisited

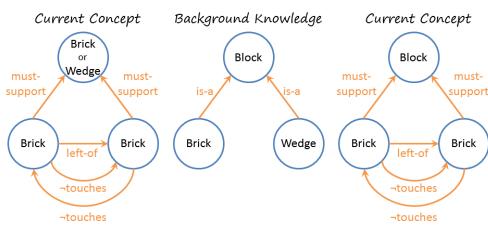


Figure 665: Incremental Concept Learning Revisited

Version spaces is a technique for learning concepts incrementally. This means that a technique is going to learn from a small set of examples, that are going to arrive one example at a time. Now we have come across a different notion of incremental concept learning earlier. In that technique, we started with a current concept definition. Then a new example will come about. In this case it's a negative example, not an arch. And then a new concept characterization would be constructed, by revising the concept characterization that we began with. The revision was such that, the new concept characterization was a specialization of the old concept characterization, such that that the example got excluded. So, negative examples led to specialization, positive examples led to generalization, and the ordering of these examples was important. We want the first example to be a positive example of the concept that we learned. We want the order to include both positive, and negative examples. That technique was very useful when there was a teacher available, who would give you these examples in the right order. Now,

the order of examples was very important in that technique of incremental concept learning. We want this set of examples to include both positive and negative examples, so that we can do generalization, and specialization. And we want each example to differ from the previous concept characterization in only one important feature. In this case, the only important difference between the new example, and the current concept characterization is the two bricks touch each other. That, the new example, the first one, the current concept characterization in exactly one feature, is important. Because it focuses the attention of the learner. It tells the learner, how to do the specialization, or the generalization, so as to include the difference, or to exclude the difference. This technique is very useful if we teachers have a label that can give the examples in a good order, in the right order, so that effective learning can occur. In that technique of incremental concept learning. We also consider the rule of background knowledge. If the current concept characterization is brick, or wedge at the node, and the learner has background knowledge which tells it that bricks, and wedges are examples of blocks, then the learner can generalize from brick, or wedge to a block here. But now we can ask two questions. What happens if these two factors? The presence of a teacher who gives examples in a particular order. And the availability of this background knowledge tells the learner exactly how far the generalize. In general, in learning, deciding how much to generalize. Is a big problem. Learners tend to either over-journalize, in which case they come to incorrect answers. Or they tend to under-journalize, in which case the answer might be correct but not very useful. If a child comes across a dog that is hooligan, furry, black, and called Buddy, and the child decides that a dog by definition is. Four legged, furry, black, and called Buddy. Then that is an example of under generalization. The conceptual characterization is correct, but not very useful because we're not transfer to any other dog. On the other hand, if the trial decides that all four legged, furry animals, are dogs, then it's an example of over generalization. Because it could also include many cats. So, the question

now becomes what would happen if the examples did not come in the good order? They came in an arbitrary order. And if this background knowledge was not available, in that case the agent would have a hard time deciding how far to generalize. Versions places this technique that are in a certain conditions, allows the agent to converge to the right generalization.

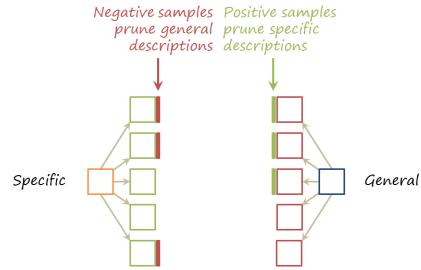


Figure 669: Abstract Version Spaces

03 - Abstract Version Spaces

[Click here to watch the video](#)

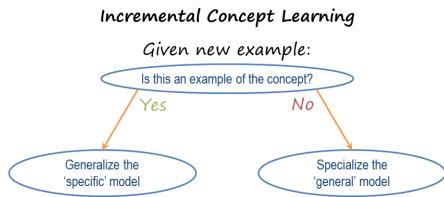


Figure 666: Abstract Version Spaces

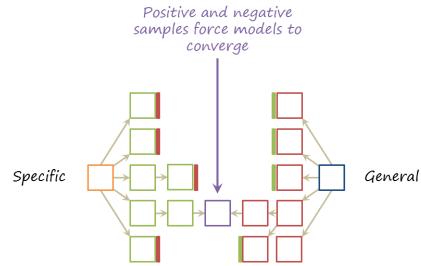


Figure 670: Abstract Version Spaces

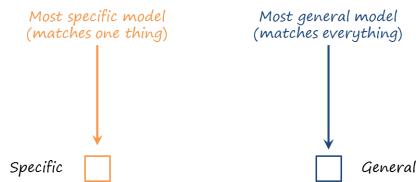


Figure 667: Abstract Version Spaces

Number	Restaurant	Meal	Day	Cost	Allergic Reaction?
Visit1	Sam's	Breakfast	Friday	Cheap	Yes
Visit2	Kim's	Lunch	Friday	Expensive	No
Visit3	Sam's	Lunch	Saturday	Cheap	Yes
Visit4	Bob's	Breakfast	Sunday	Cheap	No
Visit5	Sam's	Breakfast	Sunday	Expensive	No

Figure 671: Abstract Version Spaces

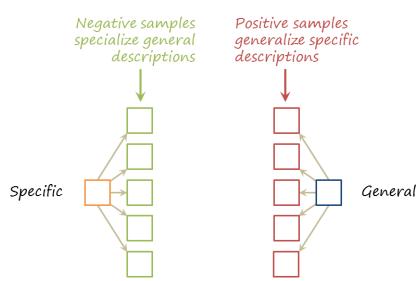


Figure 668: Abstract Version Spaces

So in the versions basis technique of learning concepts incrementally, we always have a specific model and a general model. As a new example comes along, we ask ourselves is this a positive example of the concept that we learned, or a negative example of the concept that we learned? If it's a positive example, then we generalize the specific model. If it's a negative example, we specialize the general model. Here is another set of visualizations to understand the specific and general models. This is a specific model. This is a general model. The most specific model, matches exactly one thing. The four legged, furry, black animal called Buddy. The

LESSON 19 - VERSION SPACES

most general model matches everything, all animals. Here is a current specific model, and as more positive examples come, you're going to generalize this specific model. Here are some of the generalizations. Similarly, here is the current general model, and as negative examples come, we're going to specialize the general model, and here are some of the specializations. As we'll illustrate in a minute, I'll start with the most general model and try to specialize it. Some of these generalizations and specializations that we are creating will no longer match the current data. When that happens, we'll prune them out. As we prune on this side as well on this side, the two pathways may merge depending on the ordering of the example. When they do merge, we have a solution. The right generalization of the concept for the given examples. So far we have been talking about this in very abstract terms. Let's make this concrete with an illustration.

04 - Visualizing Version Spaces

[Click here to watch the video](#)

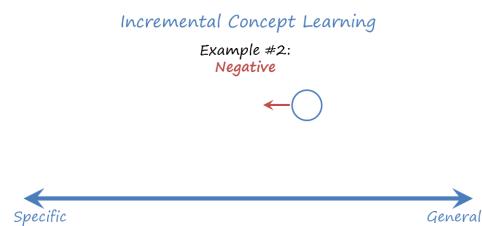


Figure 674: Visualizing Version Spaces

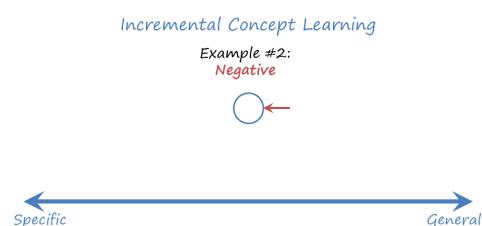


Figure 675: Visualizing Version Spaces

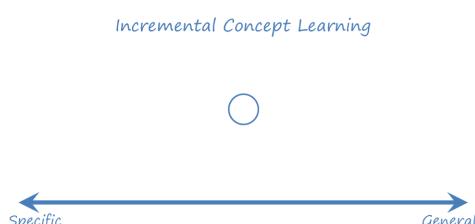


Figure 672: Visualizing Version Spaces

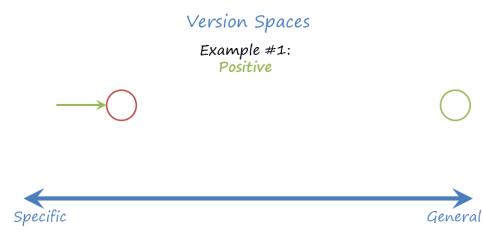


Figure 676: Visualizing Version Spaces

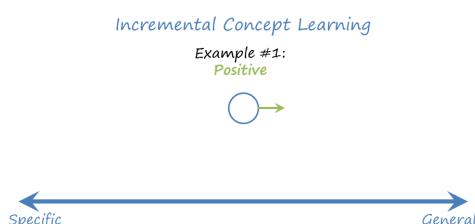


Figure 673: Visualizing Version Spaces

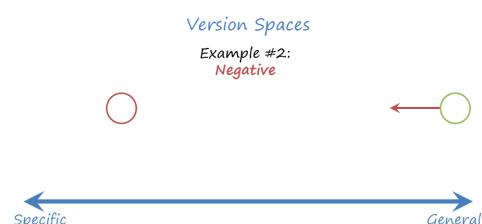


Figure 677: Visualizing Version Spaces

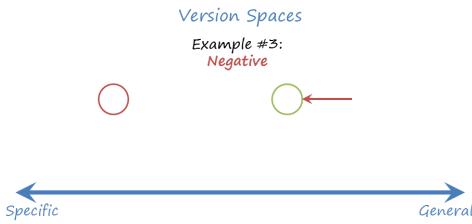


Figure 678: Visualizing Version Spaces



Figure 682: Visualizing Version Spaces

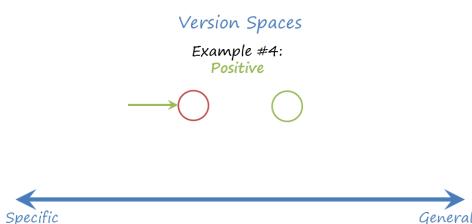


Figure 679: Visualizing Version Spaces

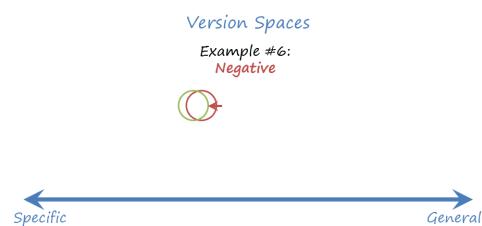


Figure 683: Visualizing Version Spaces

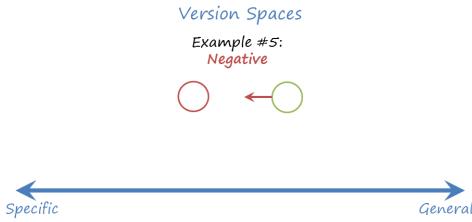


Figure 680: Visualizing Version Spaces

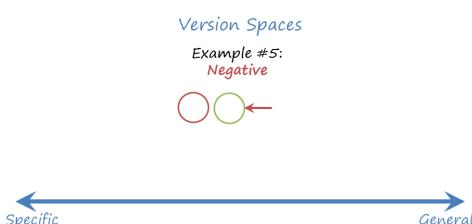


Figure 681: Visualizing Version Spaces

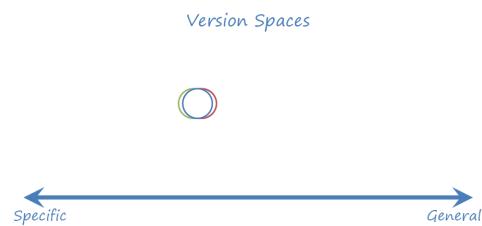


Figure 684: Visualizing Version Spaces

So Shoke, I tend to think of the difference between the incremental concept learning we've talked about in the past and version spaces in terms of a bit of a visualization. We can imagine a spectrum that runs from a very specific model of a concept to a very general model of a concept. And we can imagine that this circle represents where our model currently is. If receive a new positive example that's not currently assumed bar concept, we then generalize it a bit to move it to the right. If we receive a negative example that is currently included in our concept, we're going to move it to the left and specialize. As more and more examples come in, we start to see that our concept keeps

moving around. Notice that they would use it a model to refer to a concept. In fact, they use in concert with models almost interchangeably. This is actually quite common for certain kinds of concepts. We have discussed earlier prototypical concepts when we were discussing classification. But prototypical concepts, concepts are like models. What is a model, a model is the representation of the world. Such that there is a one-to-one correspondence what is being represented to the world and the representation itself. As an example, in the world of those blocks that made arches, I can actually make an arch in the world, and then I can build a representation of that particular arch. That's a model of the world, so the concept of an arch and the model of an arch in this particular case can be used interchangeably.

05 - Example Food Allergies I

[Click here to watch the video](#)

Number	Restaurant	Meal	Day	Cost	Allergic Reaction?
Visit1	Sam's	Breakfast	Friday	Cheap	Yes
Visit2	Kim's	Lunch	Friday	Expensive	No
Visit3	Sam's	Lunch	Saturday	Cheap	Yes
Visit4	Bob's	Breakfast	Sunday	Cheap	No
Visit5	Sam's	Breakfast	Sunday	Expensive	No

Figure 685: Example Food Allergies I

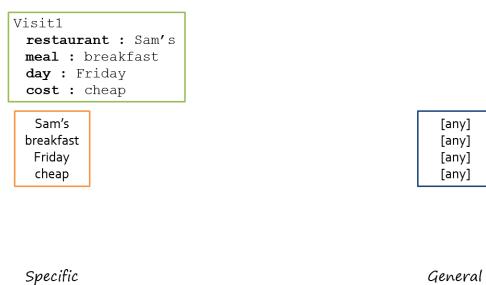


Figure 686: Example Food Allergies I

So let us suppose that I go to a number of restaurants, and have various kinds of meals and sometimes get an allergic reaction. I do not understand why I'm getting this allergic reaction,

under what conditions do I get the allergic reaction. So go to an ER agent and say, dear ER agent tell me, under what conditions do I get allergic reactions. And I give all the data, shown in this table, to the AI agent. Note that there are only five examples here, like we mentioned in knowledge-based AI we want to do learning based on a small number of examples because that's how humans do learning. Note also, that the first example is positive. And that there are both positive and negative examples. That is important so we can, construct both specializations and generalizations. How then, may an AI agent decide the conditions under which I get allergic reaction. So this examples are coming one at a time, and let us see what happens when the first example comes. Here is the first example. The restaurant was Sam's. Meal was breakfast. Day was Friday. The cost was cheap. So from this one example, I can construct both a very specific model, which is exactly this example. Sam's, breakfast, Friday, cheap. You can't have anything, more specific than this. And the AI agent can also construct a more general model. Which of course is, that it can be any restaurant, any meal, any, day and so on. You can't construct a more general model than this. So the most specialized model based on this one example says that, I'm allergic when I go to Sam's and have breakfast on Fridays and the cost is cheap. And the most general model says, I'm allergic to everything. No matter where I go, what meal I have, on what day, and what the cost is, I feel allergic.

06 - Example Food Allergies II

[Click here to watch the video](#)

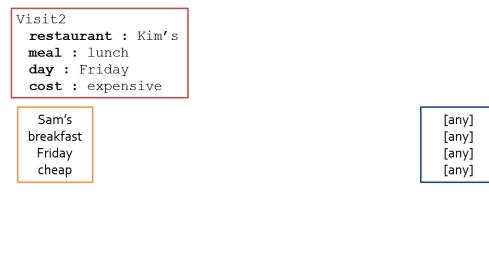


Figure 687: Example Food Allergies II

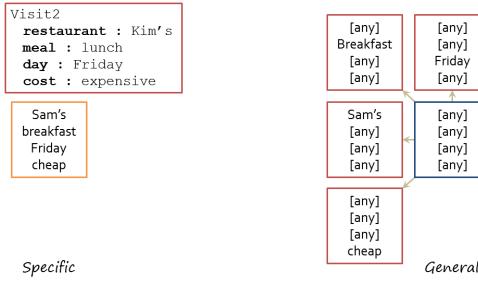


Figure 688: Example Food Allergies II



Figure 689: Example Food Allergies II

Let us consider the processing as a second example comes along. And the red outline for this example means it is a negative example. So now the agent will try to find a way of specializing the most general model and generalizing the most specialized model, in order to account for this negative example. So given this negative example, you want to specialize the most general model so that this negative example is excluded and yet each of the specializations is a generalization of this most specific model because this was coming from a positive example. We do want to include this. Let's first specialize in a way so that each specialization is a generalization of this model. There are 4 ways of doing it because there are 4 slots here. The first slot here deals with the name of the restaurant like Sam's or Kim's. One specialization of this most general concept is to put the name of an actual restaurant there. This is generalization of this concept because this was deferring to one specific need at Sam's, this is referring to any need at Sam's. In a similar way I can specialize the filler of the second slot. In short of having any meal, I can make it a breakfast meal. This is a specialization of this most general concept that

is a generalization of this concept because this refers to breakfast at any place, this refers to breakfast at Sam's on Friday and so on. Similarly for the third slot and the fourth slot in this most general concept. Now I must look at these specializations of the most general concept and ask which one of them should I prune so as to exclude the negative example. I notice that Sam's doesn't match Kim's, so this is already excluded in so far as this concept is concerned. Breakfast doesn't match lunch, so this example is already excluded as far as this concept is concerned. How about with this concept of characterization and mix this negative example, therefore I must floor it. So we pull away that particular concept characterization and we are left with three specializations of the most general model.

07 - Example Food Allergies III

[Click here to watch the video](#)

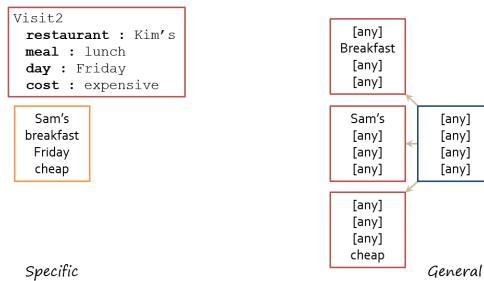


Figure 690: Example Food Allergies III

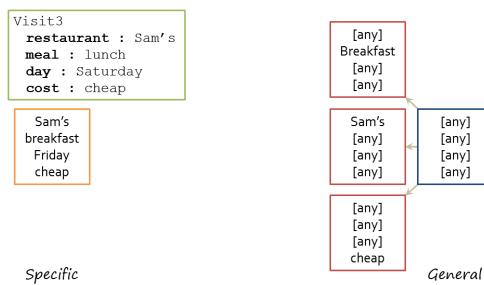


Figure 691: Example Food Allergies III

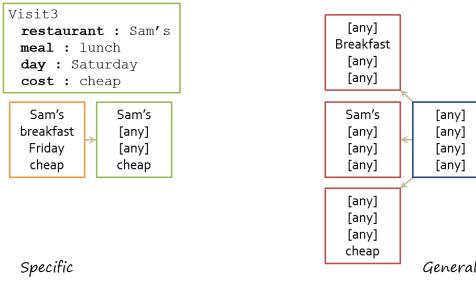


Figure 692: Example Food Allergies III

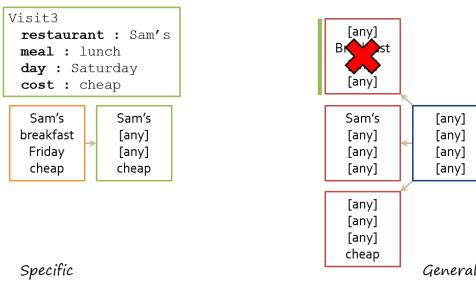


Figure 693: Example Food Allergies III

Let us consider what happens when a third example comes along. And the green outline of this example shows that this is the positive example of the concept. Because this is the positive example of the concept, we must try to generalize the most specific model. So a generalization of the specific concept, that includes this positive example as shown here. Here the meal was breakfast, here the meal was lunch. So we can generalize over any meal. Here the day was Saturday, here it was Friday, so we can generalize over any day. Of course we could have also generalized just Friday or Saturday, but for simplicity we'll generalize over any day. Similarly for breakfast or lunch, generalized to any meal. But at this stage, there is another element to the processing. We must examine all the specializations of the most general concept and see whether any one of them needs to be pruned out. The pruning may need to be done in order to make sure that each specialization here is consistent with the positive examples that are coming in. So in this case, if we look at the first specialization here, which says, I'm allergic to breakfast at any place on any day. This cannot be a generalization of this particular concept. Put another

way, there is no way that this breakfast here can include, can cover, this positive example which deals with lunch. But yet another way, the only way I can move from breakfast to any here would be if I generalize, but in this direction I can only specialize. Therefore, this must be pruned out. As you prune this first concept out, we're left with only two.

08 - Example Food Allergies IV

[Click here to watch the video](#)

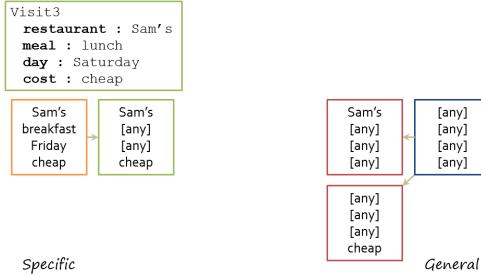


Figure 694: Example Food Allergies IV

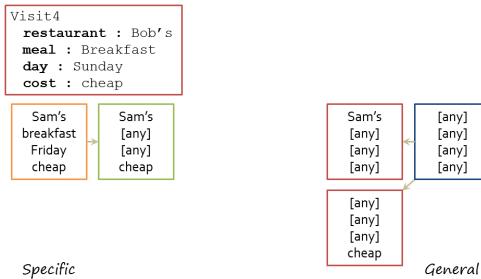


Figure 695: Example Food Allergies IV

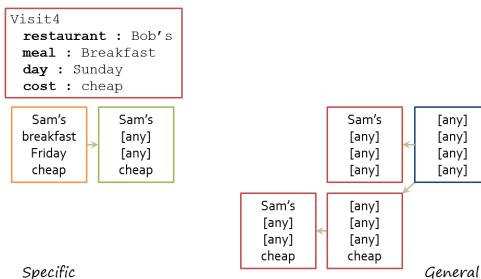


Figure 696: Example Food Allergies IV

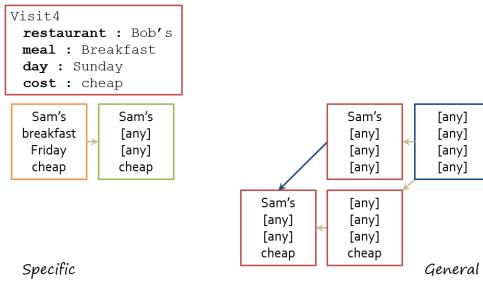


Figure 697: Example Food Allergies IV

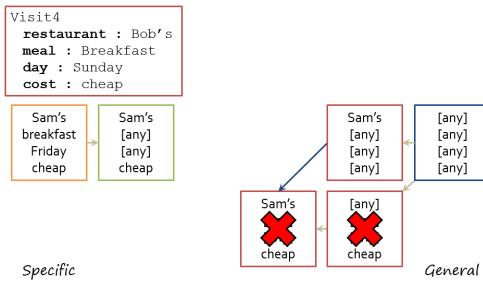


Figure 698: Example Food Allergies IV

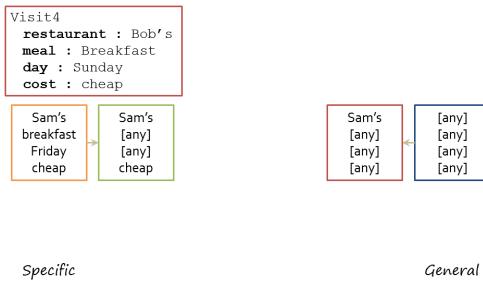


Figure 699: Example Food Allergies IV

Now let us consider, the processing of the fourth example comes along. Again the red outline shows that this is a negative example of the constant. Because this is a negative example, we must specialize in most journal concept characterizations available at the moment. We can begin by checking, whether we need to specialize this particular general concept. But wait, this general concept characterization, already excludes the negative example. This says the earlier happens when I go to Sam's, and this has Bob's in it, so this already excludes it, I don't have to specialize it any more. Now let's look at this general model. Does this need to be specialized, in order to excluded? Yes, because at

the current stage, this includes this vertical example. It is cheap here, this is cheap, this is any here, and this has particle elements within. This means that, this concept characterization, must be specialized in a way that excludes this negative example and yet. The new specialization, is consistent with the most specialized characterization at present. It is tempting to see the two pathways as converging here, because this is identical to that, but we also have this branch hanging, and this branch says that I'm allergic to any meal at Sam's, not just a cheap meal. So, we're not done yet. In this state there is one other element to consider. If there is a node, that lies on a pathway starting from the most journal concept characterization, that is subsumed by a node, that comes from another pathway starting from the same journal concept characterization, then I want to prune that particular node. The reason I wanted to put on this note is, because this note is subsumed by this note. So this note is true, I don't have to carry this around. If I'm allergic to any meat at Sam's, I don't have to specify that I'm allergic to cheap meat at Sam's, thus I can pull on this particular pathway, and I've left it only this particular pathway. At this point in processing, these are the examples that have been encountered so far. There are only two possible. I'm either allergic to everything at Sam's, or I'm allergic to every cheap meal at Sam's.

09 - Example Food Allergies V

[Click here to watch the video](#)

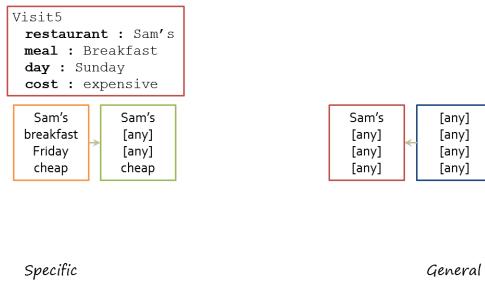


Figure 700: Example Food Allergies V

LESSON 19 - VERSION SPACES

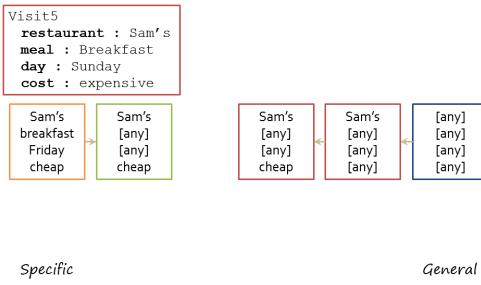


Figure 701: Example Food Allergies V

I know you' are wondering when this is going to end. We're almost done, we're almost done. Let's consider what happens when the first example comes. This is a negative example as indicated by the red outline. Because the negative example, we must specialize in most journal characterization, in such a way that this negative example is dueled out, and this specialization is consistent with. The most journal version, starting from the most specialized concept characterization. The only specialization of this journal concept, that both excludes this and is consistent with this node is, Sam's cheap. It excludes this, because it is cheap here, it will rule out the fact that this is expensive here. Now the agent noticed that these two particular consequences positions are the same and if a convergence has occurred. Now we have the answer we wanted. I get allergies whenever I go to Sam's and have a cheap meal.

10 - Version Spaces Algorithm

[Click here to watch the video](#)

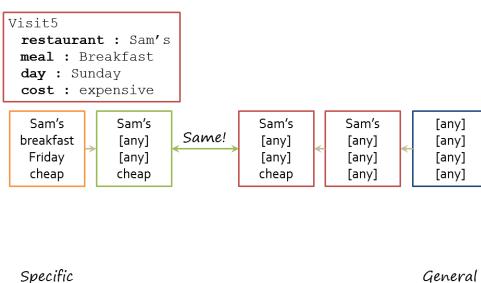


Figure 702: Version Spaces Algorithm

Algorithm for Version Spaces

For each example:

If the example is **positive**:
Generalize all **specific** models to include it
Prune away **general** models that cannot include it

If the example is **negative**:
 Specialize all **general** models to include it
 Prune away **specific** models that cannot include it

Prune away any models **subsumed** by other models

Figure 703: Version Spaces Algorithm

Number	Meal	Meal	Day	Cost	Vegan	Reaction?
Visit1	Kim's	Breakfast	Friday	Cheap	No	Yes
Visit2	Kim's	Lunch	Friday	Cheap	No	Yes
Visit3	Sam's	Lunch	Saturday	Cheap	No	No
Visit4	Kim's	Breakfast	Sunday	Cheap	Yes	No
Visit5	Sam's	Breakfast	Sunday	Expensive	Yes	No
Visit6	Kim's	Lunch	Saturday	Cheap	No	Yes
Visit7	Kim's	Lunch	Monday	Expensive	No	No

What model did you converge on?



Figure 704: Version Spaces Algorithm

What we have just done here, is a very powerful idea in learning. Convergence is important. Because without convergence, a learning agent could zig zag forever in a large learning space. We want to ensure that the learning agent converges to some concept characterization, and that remains stable. This method guarantees convergence, as long as there is a sufficiently large number of examples. We needed five examples in this particular illustration, for the convergence to occur. This convergence would have occurred, irrespective of the order of the examples, as long as the five examples were there. Note that we did not use background knowledge like we did in incremental concept learning. Note also that we did not assume that the teacher was forwarding the examples in the right order. This is the benefit of version space learning. There is another feature to note. In incremental concept learning, we wanted each example different from the current concept characterization in exactly one feature, so that the learning agent could focus its attention. However inversion spaces, you can notice that each successful example, the first one, the previous one and many features, just look at the first two examples. They differ in

many features in the name of the restaurant, in the meal, in the cost. Here is the algorithm for the version space technique. We'll go through it very quickly, because we've already illustrated it in detail. If the new example is positive, generalize all specific models included. Prune away the general models that cannot include the positive example. If the example is negative, specialize all the general models to include it. Prune away the specific models that cannot include the negative example. Prune away any models subsumed by the other models. Know that in this specific implementation of version space technique that we just illustrated, there is a single pathway coming from the most specialize concert model. And therefore there is no need to prune away specific models. In general, there could be multiple generalizations coming for the most specialized models, and this might be needed.

11 - Exercise Version Spaces I

[Click here to watch the video](#)

Visit1
restaurant : Kim's meal : Breakfast day : Friday
cost : Cheap vegan : no



What would the initial general and specific models be?

Figure 705: Exercise Version Spaces I

Let us do some exercises together. This exercise actually is quite similar to the exercise we had done previously except that we have added one more feature, vegan. Either the meat can be vegan or the meat is not vegan. Suppose this is the first example that comes along and this is the positive examples indicated by the green outline. Write down the most specific and the most general models

12 - Exercise Version Spaces I

[Click here to watch the video](#)

Visit1
restaurant : Kim's meal : Breakfast day : Friday
cost : Cheap vegan : no



What would the initial general and specific models be?

Figure 706: Exercise Version Spaces I

So, this example is pretty similar to the case we had in the previous example. So, the most specific case is that I'm simply allergic to any breakfast that comes on Friday that's cheap and isn't vegan, so this very specific example. And the most general model is I'm just allergic to everything, no matter what meal it is, what day it is, how much it costs, whether it's vegan, or what restaurant I got it at.

13 - Exercise Version Spaces II

[Click here to watch the video](#)

Visit2
restaurant : Kim's meal : Lunch day : Friday
cost : Cheap vegan : no



Based on this example, would we generalize or specialize?

- Generalize
- Specialize

Figure 707: Exercise Version Spaces II

Now suppose a second example comes along, and this example is also positive as indicated by the green outline. Based on the second example, would you specialize or would you generalize?

14 - Exercise Version Spaces II

[Click here to watch the video](#)

LESSON 19 - VERSION SPACES

Visit2
 restaurant : Kim's meal : Lunch day : Friday
 cost : Cheap vegan : no

Kim's
Breakfast
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

Based on this example, would we generalize or specialize?

- Generalize
- Specialize

Visit2
 restaurant : Kim's meal : Lunch day : Friday
 cost : Cheap vegan : no

Kim's
Breakfast
Friday
Cheap
no

Kim's
[any]
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

After generalizing, what will the general model be?

Figure 708: Exercise Version Spaces II

Figure 710: Exercise Version Spaces III

What do you think they would generalize or specialize? So we said earlier that whenever we get a positive example then we generalize our most specific model. So we need to generalize this model based on this new example. It's no longer sufficient to say I'm just allergic to any breakfast at Kim's and I find that it's cheap and not vegan. There's something else I'm allergic to. That's right, David.

What did you write down, David? So, in this case, the only difference between this and the previous positive example is that now it's lunch instead of breakfast. So, the only thing we're going to generalize here is that now it can be any meal instead of just being breakfast, but we're still going to say that it has to be Kim's on Friday. It has to be cheap, and it can't be vegan. And note that they could have put here breakfast or lunch, but for simplicity has generalized it to any meal.

15 - Exercise Version Spaces III

[Click here to watch the video](#)

Visit2
 restaurant : Kim's meal : Lunch day : Friday
 cost : Cheap vegan : no

Kim's
Breakfast
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

After generalizing, what will the general model be?

Visit3
 restaurant : Sam's meal : Lunch day : Saturday
 cost : Cheap vegan : no

Kim's
Breakfast
Friday
Cheap
no

Kim's
[any]
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

Based on this example, would we generalize or specialize?

- Generalize
- Specialize

Figure 709: Exercise Version Spaces III

Figure 711: Exercise Version Spaces IV

So write down of the generalization of this most specific model that is consistent with this positive example.

Let's go a little bit further, suppose a third example comes along, and this is the negative example indicated by the red outline here. What would you do this time? Generalize or specialize?

16 - Exercise Version Spaces III

[Click here to watch the video](#)

18 - Exercise Version Spaces IV

[Click here to watch the video](#)

```
Visit3
restaurant : Sam's    meal : Lunch    day : Saturday
cost : Cheap          vegan : no
```

Kim's
Breakfast
Friday
Cheap
no

Kim's
[any]
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

Based on this example, would we generalize or specialize?

Generalize

Specialize

```
Visit3
restaurant : Sam's    meal : Lunch    day : Saturday
cost : Cheap          vegan : no
```

Kim's
Breakfast
Friday
Cheap
no

Kim's
[any]
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

How many potential general models will we have after specializing based on this case and pruning?

3

Figure 712: Exercise Version Spaces IV

Figure 714: Exercise Version Spaces V

So, this time we're going to specialize our most general model. It's obvious that I'm not allergic to absolutely everything everywhere, because here's a particular instance where I wasn't allergic to what I ate. So we're going to specialize our most general model.

19 - Exercise Version Spaces V

[Click here to watch the video](#)

```
Visit3
restaurant : Sam's    meal : Lunch    day : Saturday
cost : Cheap          vegan : no
```

Kim's
Breakfast
Friday
Cheap
no

Kim's
[any]
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

How many potential general models will we have after specializing based on this case and pruning?

Figure 713: Exercise Version Spaces V

So like David said, given this negative example, we'll specialize this most general model. And we'll prune out those specializations that no longer match the data. Given this, how many specializations are left after the pruning?

20 - Exercise Version Spaces V

[Click here to watch the video](#)

```
Visit3
restaurant : Sam's    meal : Lunch    day : Saturday
cost : Cheap          vegan : no
```

Kim's
Breakfast
Friday
Cheap
no

Kim's
[any]
Friday
Cheap
no

[any]
[any]
[any]
[any]
[any]

[any]
[any]
[any]
[any]
[any]

Figure 715: Exercise Version Spaces V

So I said that there'll be three potential general models left after specializing and pruning. Those three models are going to be that I could just be allergic to everything at Kim's, I could just always be allergic to breakfast, or I could just be allergic to eating on Friday. I would prune the ones based on cost and whether or not the meal is vegan, because although I've had bad reactions to cheap, non-vegan meals in the past, here I didn't have a reaction to a cheap, non-vegan meal. So it's not sufficient to say I'm allergic to everything non-vegan or I'm allergic to all cheap food.

21 - Exercise Version Spaces VI

[Click here to watch the video](#)

Number	Meal	Meal	Day	Cost	Vegan	Reaction?
Visit1	Kim's	Breakfast	Friday	Cheap	No	Yes
Visit2	Kim's	Lunch	Friday	Cheap	No	Yes
Visit3	Sam's	Lunch	Saturday	Cheap	No	No
Visit4	Kim's	Breakfast	Sunday	Cheap	Yes	No
Visit5	Sam's	Breakfast	Sunday	Expensive	Yes	No
Visit6	Kim's	Lunch	Saturday	Cheap	No	Yes
Visit7	Kim's	Lunch	Monday	Expensive	No	No

What model did you converge on?

Kim's
[any]
[any]
Cheap
[any]

[any]
[any]
[any]
Lunch
Friday
[any]
No

Kim's
[any]
[any]
Lunch
Friday
Cheap
No

Kim's
[any]
[any]
Cheap
No

[any]
[any]
[any]
Cheap
No

Figure 716: Exercise Version Spaces VI

LESSON 19 - VERSION SPACES

We'd like you to complete this exercise. We've already done the first three examples. Having completed the exercise, decide which model you converge on.

22 - Exercise Version Spaces VI

[Click here to watch the video](#)

Number	Meal	Meal	Day	Cost	Vegan	Reaction?
Visit1	Kim's	Breakfast	Friday	Cheap	No	Yes
Visit2	Kim's	Lunch	Friday	Cheap	No	Yes
Visit3	Sam's	Lunch	Saturday	Cheap	No	No
Visit4	Kim's	Breakfast	Sunday	Cheap	Yes	No
Visit5	Sam's	Breakfast	Sunday	Expensive	Yes	No
Visit6	Kim's	Lunch	Saturday	Cheap	No	Yes
Visit7	Kim's	Lunch	Monday	Expensive	No	No

What model did you converge on?

Kim's	[any]	[any]	[any]	[any]	[any]	[any]
[any]						
Cheap	[any]	[any]	[any]	[any]	[any]	[any]
[any]						

[any]	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
Lunch	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
Friday	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
No	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]

Kim's	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
Lunch	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
Friday	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
No	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]

Kim's	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
Lunch	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
Friday	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]
No	[any]	[any]	[any]	[any]	[any]	[any]
[any]	[any]	[any]	[any]	[any]	[any]	[any]

Figure 717: Exercise Version Spaces VI

Note that in this exercise, there were only seven examples and only five features. So we could do it by hand. What would happen if the number of examples was much larger and the number of features were much larger? This algorithm would still work but we'll need a lot more computing power. It is also possible that the algorithm may not be able find the right concept to converge to because I might be allergic to multiple meals at multiple restaurants such as breakfast at Kim's and lunch at Sam's. But even in that case, the benefit of this algorithm is it will show that convergence is not possible even after many, many examples.

23 - Identification Trees

[Click here to watch the video](#)



Figure 718: Identification Trees

Number	Restaurant	Meal	Day	Cost	Allergic Reaction?
Visit1	Sam's	Breakfast	Friday	Cheap	Yes
Visit2	Kim's	Lunch	Friday	Expensive	No
Visit3	Sam's	Lunch	Saturday	Cheap	Yes
Visit4	Bob's	Breakfast	Sunday	Cheap	No
Visit5	Sam's	Breakfast	Sunday	Expensive	No

Figure 719: Identification Trees

Number	Restaurant	Meal	Day	Cost	Allergic Reaction?
Visit1	Sam's	Breakfast	Friday	Cheap	Yes
Visit2	Kim's	Lunch	Friday	Expensive	No
Visit3	Sam's	Lunch	Saturday	Cheap	Yes
Visit4	Bob's	Breakfast	Sunday	Cheap	No
Visit5	Sam's	Breakfast	Sunday	Expensive	No



Figure 720: Identification Trees

Number	Restaurant	Meal	Day	Cost	Allergic Reaction?
Visit1	Sam's	Breakfast	Friday	Cheap	Yes
Visit2	Kim's	Lunch	Friday	Expensive	No
Visit3	Sam's	Lunch	Saturday	Cheap	Yes
Visit4	Bob's	Breakfast	Sunday	Cheap	No
Visit5	Sam's	Breakfast	Sunday	Expensive	No



Figure 721: Identification Trees

It is one of the methods we can use, to process the kind of the data that we just saw. It is sometimes called decision-free learning. Recall that we were discussing case-based learning, we talked about discrimination tree learning. There, we learned the discrimination tree incrementally. A case would come one at a time, and we would ask the question, what feature would discriminate between the existing cases, and the new case? And we would pick a feature. Discrimination pre-learning provides no guarantee of the optimality of this tree. That is to say, at retrieval time, when a new problem comes along, traversing this tree might take a long time because this tree is not the most optimal tree was

during these cases. We'll discuss an alternative method called decision tree learning, which will give us more optimal trees, however, at a cost. The cost will be that all the examples will need to be given right at the beginning. Let us return to our restaurant example. We want to learn a decision tree that will classify these five examples so that as a new problem comes along, we can quickly find which is the closest example to the new problem. To do this, we need to pick one of four features, restaurant, meal, day or cost that will separate these allergic reactions, so that one category contains either only false instances, or only true instances. As an example, supposing we think of restaurant as being the decisive feature. So we have picked restaurant as a decisive feature. Now, there are three kinds of restaurants. Kim's, Bob's, and Sam's. Whenever it's Kim's restaurant, or Bob's restaurant, there is no allergic reaction. Whenever it's Sam's restaurant, there can be allergic action shown in green here, or no allergic reaction, shown in red. So the good thing about this particular feature, restaurant, is that, it has separated all the five examples into two classes. Into the class Sam's, and into the class not Sam's. Not Sam class consists of only negative reactions, which is good, because we know that we have now been able to classify all of these five examples into two sets, one of which contains only negative examples. Now for these three examples, you must pick another feature that will separate them into positive and negative instances. In this case, we might consider cost to be that feature. When the cost is cheap, then we get positive examples. When the cost is expensive, then we get negative examples. This is a classification tree. And in fact, this is a very efficient classification tree. When a new problem comes around, for example visit6. Sam's, lunch, Friday, cost is expensive, and you want to decide what the allergic reaction might be, we simply have to travel through this tree, to find out, the closest neighbor, of that particular new example. This is called a decision tree. And this technique that we just discussed is called decision tree learning. This method of inductive decision tree learning worked much more efficiency and apparently more easily than earlier

method that we have discussed. But the trade off is that we needed to know all the five examples right in the beginning. Of course, this technique simply appears to be efficient and easy. And that is because we had only five examples, and only four features that were describing all five examples. If the number of examples was very large, or the number of features that were describing the examples were very large. Then it's very hard to decide what exactly should be the feature that we should use to discriminate on.

24 - Optimal Identification Trees

[Click here to watch the video](#)

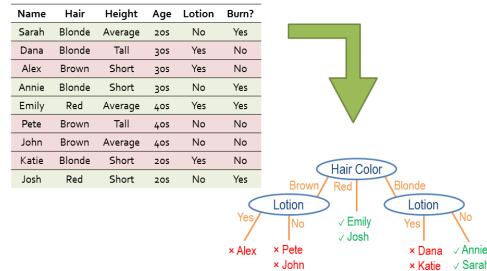


Figure 722: Optimal Identification Trees

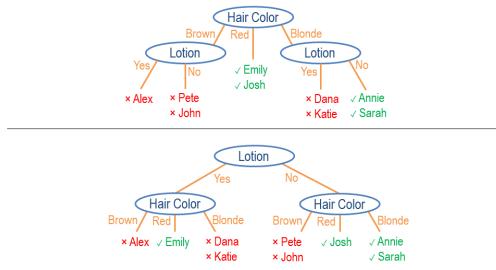


Figure 723: Optimal Identification Trees

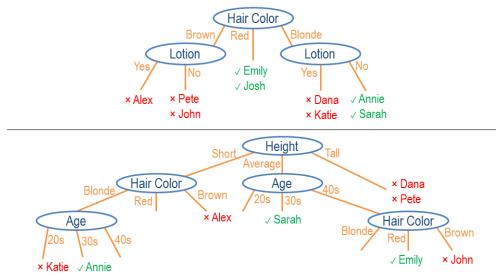


Figure 724: Optimal Identification Trees

Let us look at another example of decision tree learning. Here is a data set of people who go to the beach, and some of them get sunburned, and others don't. In this data set, there are nine examples and each example is characterized by four features, hair, height, age and lotion. Once again, how can we construct an optimal decision tree that they classify all of those examples? One possible idea is to discriminate first on hair color. Hair color classifies all of these known examples into three categories, brown, red and blonde. The interesting thing about the choice of hair color is that in the case of brown, all of these sunburnt cases are negative. People with brown hair apparently don't get sunburned. In case of all the red haired people, there is sunburn. So hair color is a good choice for picking as a feature to discriminate on because it classifies things in such a way that some of the categories have only negative instances and no positive instances. And some of the categories are only positive instances and no negative instances. Of course, that still leaves blonde-haired people. In this case, there are both some positive instances and some negative instances, and therefore, will need another feature to discriminate between the positive and the negative instances. Here, lotion might be the second feature that we pick. Lotion now classifies the remaining examples into two categories, some people used lotion, other people did not. Those who used lotion did not get sunburnt. Those who did not use lotion did get sunburn. Once again, these are all negative instances. These are consisting of only positive instances. Thus, in this decision tree, simply by using two features, we were able to classify all of these nine examples. This is a different decision tree for this same data set. But because we use a different order, therefore, now we have to do more work. This decision tree is less optimal than the previous one. We could have chosen a different set of features in a different order. Perhaps, we could first discriminate on height then on hair color and age. In this case, we did a much bushier tree. Clearly, this tree is less optimal than this one. Note the trade off with the decision tree learning and discrimination tree learning that we covered in case-based reasoning.

Decision tree learning leads to more optimal classification trees. But there is a requirement. You need all the examples right up front. Discrimination tree learning may lead to suboptimal trees, but you can learn incrementally.

25 - Assignment Version Spaces

[Click here to watch the video](#)

Assignment

How would you use version spaces to design an agent that could answer Raven's progressive matrices?

Figure 725: Assignment Version Spaces

So how would version spaces be useful to answer Raven's progressive matrices? Like with the incremental concept learning, think first about what concept you're trying to learn. Are you learning transformations? Are you learning types of problems? What are the increments? Are they individual problems? Are they individual figures? Are they individual transformations in a problem? Second, what are you converging onto? For example, you could use version spaces within one problem and converge down onto a correct answer, or you could use it for learning how to solve problems in general and converge onto adoptable algorithm, or you could use it for learning an ontology of problems and converge onto a single type of problem you expect to see in the future. So what are you converging onto if you use version spaces for Raven's progressive matrices?

26 - Wrap Up

[Click here to watch the video](#)

- To recap...
- Definition of version spaces
 - Algorithm for version spaces
 - Complex problems with version spaces
 - Limitations and questions
 - Identification trees

Figure 726: Wrap Up

So today we've talked about version spaces. Version spaces are an algorithm for converging onto an understanding of a concept, even in the absence of prior background knowledge or an intelligent teacher. We covered the algorithm for version spaces, where we iteratively refine a general and specific model of a concept, until they converge down onto one another. We then talked about using version spaces to address more complex problems. We've also connected version spaces to older concepts, like incremental concept learning. Finally we talked about the limitations of version spaces, such as what to do if there's no single correct concept, or what to do in the absence of either positive or negative examples. To address these, we also covered identification trees, which are a different ways of approaching the same kind of data that version spaces operate on. We'll touch on version spaces and incremental concept learning again, when we talk about mistake based learning.

27 - The Cognitive Connection

[Click here to watch the video](#)

Cognitive agents too face the issue of how far to generalize. We can undergeneralize in which case what we learn is not very useful. We can overgeneralize in which case what we learned may not be correct. For example, imagine that I was a Martian who came to your Earth. I saw the first human being, and I may undergeneralize and say this specific person has two arms. That is not very useful because that is not applicable to any other human being. Or I may overgeneralize and say everyone on this Earth has

two arms. That may not be correct. [Version Spaces] is a technique that allows convergence to the right level of abstraction. This is also connected to the notion of cognitive flexibility. Cognitive flexibility occurs where the agent has multiple characterizations or multiple perspectives on the same thing. As we saw in version spaces, the agent has several possible definitions for a concept that converge over time. An alternate view is to come up with one generalization and try it out in the world. See how well it works. If it leads to a mistake or a failure, then one can learn by correcting that mistake. We'll return to this topic a little bit later in the class.

28 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

29 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

Version Spaces is a technique for learning concepts incrementally. In other words, Version spaces are an algorithm for converging onto an understanding of a concept, even in the absence of prior background knowledge or an intelligent teacher. The algorithm for Version spaces iteratively refines a general and specific model of a concept, until they converge down onto one another.

References

Optional Reading:

1. Winston Chapter 20; [Click here](#)

Exercises

None.

Lesson 20 - Constraint Propagation



The more constraints one imposes, the more one frees one's self. And the arbitrariness of the constraint serves only to obtain precision of execution.
– Igor Stravinsky: Russian music composer.

01 - Preview

[Click here to watch the video](#)

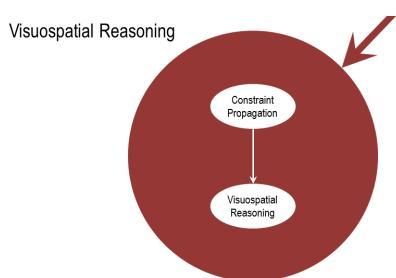


Figure 727: Preview

AI, and there are a number of different topics, such as planning, understanding, natural language processing, visual spatial reasoning. Today, we'll start by defining constraint propagation. Then we'll see how it helps agents make sense of the world around it. Our examples will come mostly from understand natural language sentences, as well as visual scenes. Finally, we'll talk about some advanced issues of constraint propagation.

Lesson Preview

- Definition
- Image processing
- Natural language understanding
- Advanced problems

02 - Exercise Recognizing 3D Figures

[Click here to watch the video](#)

Figure 728: Preview

Today, we'll talk about constraint propagation, another very drawn out purpose method. Constraint propagation is a mechanism of influence where the agent assigns values to variables to satisfy certain conditions called constraints. It is a very common method in knowledge-based

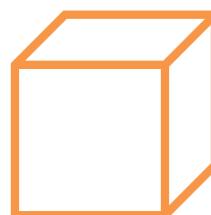


Figure 729: Exercise Recognizing 3D Figures

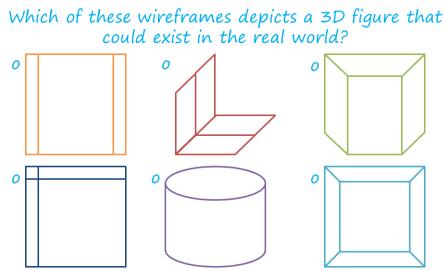


Figure 730: Exercise Recognizing 3D Figures

To illustrate the technique of constraint propagation, let us consider this figure drawn on a 2D surface. Although this figure has been drawn on a 2D surface, you and I can almost immediately recognize that this is a 3D cube. How did we recognize that this is a 3D cube? How can we help machines do it? Cube is an example of a trihedral object. A trihedral object is one with three surfaces joined at a particular point, at this particular point. In general, what [one can have] a polyhedral surfaces. A polyhedral surface is one where multiple surfaces join at the same point. So a soccer ball, with its white and black patterns, is an example of a polyhedral object because at one point several surfaces can join. A pyramid is another good example of a polyhedral object. A pyramid has four surfaces joining, at the apex. So let's do a simple exercise together. Here are six figures all drawn on a 2D surface. Which of these six figures do you think represents a 3D object?

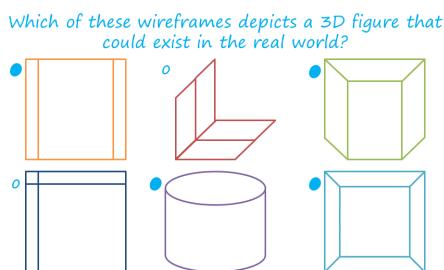
03 - Exercise Recognizing 3D Figures[Click here to watch the video](#)

Figure 731: Exercise Recognizing 3D Figures

What do you think, David? So I said that four of them could represent a 3D object. This

could be a cylinder. This is some kind of trapezoidal prism, and this kind of looks like a pyramid from above with the top blocked off. We gotta see this as looking down at some kind of pyramid or we might also see this as looking into a box. This one is a bit of an interesting example. It doesn't as naturally lend itself to interpretation as a 3D figure. But at the same time, we can imagine that that's what you would get if you looked at this figure directly down. David, it is interesting that you did not pick this as an answer, as well. I guess I would have picked that one. I see this as an x-axis, the y-axis, and the z-axis. And because all three axis can be identified here, I would have thought of this, too, as an example of a 3D object. That's interesting, Ashok. Now that you mention it, I can't stop seeing it as a 3D object. Now I see it as kind of a folded piece of paper that just has a line drawn at the center. Initially, I don't think I looked at that as a 3D object, because I saw the lines as having to represent places where the figure folded or where different planes met. But if we look at the line at the middle as just kind of something drawn on the figure, it immediately pops out as a 3D object for me. And this brings us to the point of this exercise. The point of the exercise is that clearly some kind of processing is agreeing in our visual system. That allows us to group these lines and these surfaces in ways. So that we can identify which one of them is a 3D object, and which one of them is not a 3D object. Clearly, this processing is not completely definitive, in that sometimes it is ambiguity. You might come up with one answer to this, and someone else may come up with a slightly different to this because the processing leaves room for ambiguity.

04 - Exercise Gibberish Sentences[Click here to watch the video](#)

Which of these sentences are grammatically correct?

- Colorless green ideas sleep furiously.
- Soft drinks due from thank bills insurance.
- The treating physician persons the following excluded tasty.
- Wall decor notifies business cards of nonsensical whims.
- Tuesday brought a sharp-edged suite of pumpernickel.
- Go search or revoke to writing this present understand.

Figure 732: Exercise Gibberish Sentences

To look more deeply into the processing that might be a [occurring in our] visual system, that allows us to identify which objects are 3D objects and which ones are just 2D. Let us consider a different example. Shown here are six sentences. None of the sentence makes much sense semantically. Nevertheless, some of the sentences are grammatically correct. And you and I can quickly detect which of the sentences are grammatically correct. Can you identify which of the sentences are grammatically correct?

05 - Exercise Gibberish Sentences

[Click here to watch the video](#)

Which of these sentences are grammatically correct?

- Colorless green ideas sleep furiously.
- Soft drinks due from thank bills insurance.
- The treating physician persons the following excluded tasty.
- Wall decor notifies business cards of nonsensical whims.
- Tuesday brought a sharp-edged suite of pumpernickel.
- Go search or revoke to writing this present understand.

Figure 733: Exercise Gibberish Sentences

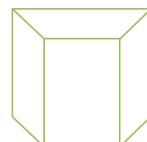
Which ones did you think, David? So the first sentence, colorless green ideas sleep furiously. I recognized pretty quickly as a pretty famous example proposed by Noam Chomsky of how sentences can be semantically nonsensical, but still grammatically correct. The sentence makes sense. There's still a subject, a verb, an adverb describing the verb, but the idea of a colorless green idea makes no sense. The idea of ideas sleeping makes maybe a little bit of sense. The idea of sleeping furiously is kind

of nonsense. So the sentence really doesn't have much meaning even though it is grammatically correct. The second sentence, soft drinks due from thank bills insurance really doesn't make any grammatical sense. It does have a seeming subject and a seeming verb. But the verb comes right a preposition suggesting that the verb fills in the slot proposed by the preposition, which doesn't make any sense. So this one doesn't make any grammatical sense to me. The same can be said for the other two negative examples. They don't really have the normal subject, predicate sentence structure or they have other rules or constraints that are being violated. For the other two positive examples, they still maintain this subject verb sentence structure. So here we have wall decor notifies, which doesn't make sense but it matches our structure. And here we have Tuesday brought something, which also makes no semantic sense, but can still be thought of as grammatically correct. Similarly, Tuesday bringing something does make some sense, but a a sharp-edged suite of pumpernickel makes no sense whatsoever. Note the vocabulary that David used in trying to find out which of these sentences were grammatically correct. It seemed to me that he was examining if the structure of these sentences was fulfilling certain conditions or fulfilling certain constraints that he expects from his knowledge of English-language grammar. One could even say that he was doing constraint processing.

06 - Constraint Propagation Defined

[Click here to watch the video](#)

Constraint propagation: a method of inference that assigns values to variables characterizing a problem in such a way that some conditions (called constraints) are satisfied.



Colorless green ideas sleep furiously.

Figure 734: Constraint Propagation Defined

This brings us to the definition of constraint propagation. Constraint propagation is a

method of inference that assigns values to variables characterizing a problem in such a way that some conditions are satisfied. So if you have any problem, that problem is going to be characterized by some variables. And the task is to give specific values to each variable in such a way that some global constraints have been satisfied. So given a problem, some problems are characterized by a set of variables, each variable may locally take on some values. So probably the question is, how can we locally assign values to variables in such a way that some global constraints are satisfied? As an example, let us return to this figure. Here is a figure, drawn on a 2D surface, and the problem is whether or not it represents a 3D object. The variables here are the surfaces and the orientations. One could consider this to be a single, two dimensional surface with some lines drawn on it. Alternatively, one can think of this as having four surfaces, one, two, three, four, where each surface has a particular orientation. The orientation can be specified by the perpendicular of that surface. The method of constraint propagation is going to help identify the surfaces and their orientations. The constraints here are defined by these junctions. For trihedral objects where three surfaces meet at a particular point, these junctions have certain properties. No matter how we assign these surfaces and their orientations, the assignment must satisfy all of those constraints. We'll look at the details of constraint propagation in a minute. But first notice that there are two possible interpretations to this particular 3D object. One can look at it as if one were looking inside a box. Alternatively, one can look at it as if one were looking at a building. This means that constraint propagation need not necessarily always succeed in this ambiguity between different kind of assignments of surfaces in the orientation. Sometimes multiple interpretations can simultaneously satisfy all the constraints. It is also possible that no assignment of values with variables will satisfy all the constraints, in which case interpretation becomes very difficult. As another example, let us examine this sentence. Colorless green ideas sleep furiously. All of us can recognize that this is semantically meaningless, but grammatically

correct. How did we know that this is grammatically correct? The variables here are the various lexical categories, like words and nouns and some different predicates. The values are the assignments we make to these various words here. Is green a noun? Is green a verb? Green a determiner? Is it part of a subject or part of a predicate? The constraints here are defined with the rules of English language grammar. As we assign values to the various variables here, that assignment must satisfy the constraints of the English language grammar so that as we assign values to these variables, those assignments must satisfy the constraints of the English language grammar before we can accept this sentence to be grammatically correct. If this sentence was grammatically not correct, then we will not be able to assign values to all the variables in a way that will satisfy the constraints imposed by the English language grammar. So we've actually come across this idea of constraints in English language grammar before. During our lesson on understanding we talked about how a preposition, for example, can constrain the meaning of the word that follows it. If we see the word from, for example, we expect what comes after it to be some kind of source for the sentence. There we used grammatical constraints in service of some kind of semantic analysis. Here we're just using grammatical constraints to figure out if a sentence is grammatically correct or not. There's another connection here to understanding as well. Ashok talked about how we can interpret this shape as either popping out towards us or down into the screen. We talked about two simultaneously accurate interpretations of the same thing and understanding with regard to sentences that can be read as puns. So for example, when I said, it's hard to explain puns to kleptomaniacs because they always take things literally. The word take can simultaneously be interpreted as interpret and physically remove, while satisfying all the constraints of the sentence.

07 - From Pixels to 3D

[Click here to watch the video](#)

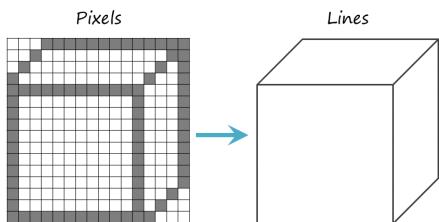


Figure 735: From Pixels to 3D

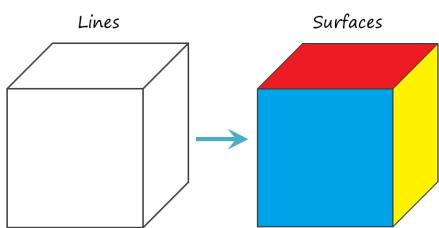


Figure 736: From Pixels to 3D

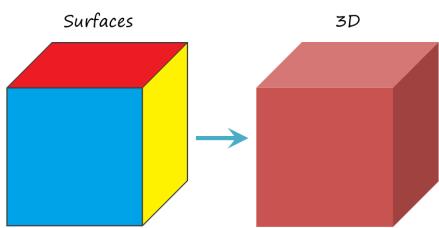


Figure 737: From Pixels to 3D

Let us look at the details of constraint propagation. To do so, we'll take a specific example from computer vision. Here's an example of a 2D image composed of a large number of pixels. The greyness at any one pixel is a depiction of the intensity of light at that pixel. Now of course, you can immediately recognize that this is a cube. But how do we do it, and how can we make a machine do it? [Miles] decompose a task of 3D object recognition into several several sub-tasks. Miles said in the first sub-task, a visual system detects edges, or lines as shown here. At this particular point, no surfaces have been detected. In this particular point, no 3D object has been finalized. Just these pixels have

been put into lines based on the intensities of light in different pixels. According to Miles the second sub task of object recognition consists of grouping these lines and the surfaces with orientations, as indicated here. So now these four lines have been grouped into the surface, and then orientation defined by the perpendicular the surface, and similarly these four lines, and these four lines. In the third and final phase of the object recognition task, according to Miles surfaces are grouped into a complete 3D object. At this particular point, your visual system recognizes that this is a cube. Miles theory has been very influential in computer vision. It has actually also been influential in AI as a whole. One of the lessons we can take away from Miles' theory of computer vision of object's recognition is that before we get into our guarded tones for addressing the task, we want to understand how a task gets decomposed into sub tasks. Throughout this course, we have emphasized task decomposition repeatedly. As an example, when we were talking about understanding, a big task of understanding got decomposed into a series of small tasks. Where surface level cues acted as probes into memory and a frame was retrieved. The slice of the frames dented expectations. Lexicon and grammatical analysts led to the identification of objects and predicates that would satisfy those expectations. And the fillers were put in. Problem reduction certainly is a general purpose method for decomposing complex tasks into smaller tasks. This notion of class decomposition is a powerful idea irrespective of what algorithm we use for any of these specific sub tasks

08 - Line Labeling

[Click here to watch the video](#)

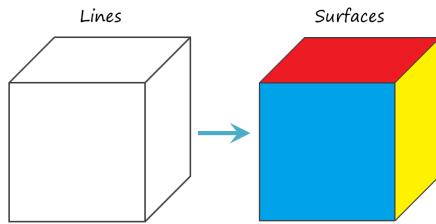


Figure 738: Line Labeling

Of course, I will focus here is on constraint propagation, not on computive vision, we are simply using some aspect of computive vision to illustrate constraint propagation. In particular, let us zoom into a specific subtask of object recognition, In this subtask, lines are grouped into surfaces and the orientations of the surfaces are identified via the perpendiculars. The method we'll use for this task is called, line labelling, the method of line labelling makes extensive use of constraints.

09 - Constraints Intersections and Edges

[Click here to watch the video](#)

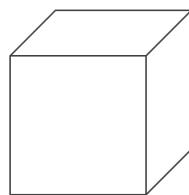


Figure 739: Constraints Intersections and Edges

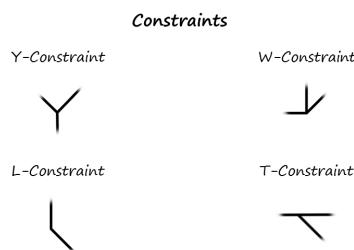


Figure 740: Constraints Intersections and Edges

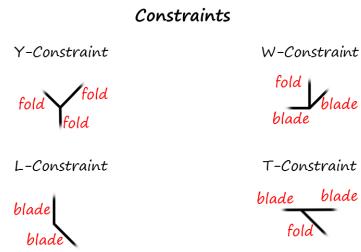


Figure 741: Constraints Intersections and Edges

So let's take the notion of constraints. Consider this cube again. You'll notice this cube has junctions, and these junctions have different kind of shapes. For example, this looks like a Y junction, this looks like an L junction, this also looks like an L junction, it's just that this arm of the L is coming in the other direction. This also looks like an L junction. This junction, on the other hand, looks a little bit like a W junction. So, junctions of various kinds. Here are the kind of junctions that can occur, in the world of trihedral objects like cubes. Y junction, W junction, T junction, L junction. We can say a little bit more about each of these junctions. Let us look at the Y junction first. If we examine the various kinds of Y junctions that get formed in the world of trihedral optics, then we find that whenever there is a Y junction formed, then each of these lines represents a fold, where a fold is a line where two surfaces meet. Now, the important thing about this is. That if we can infer, that this is a Y junction and that this line represents a fold, then an image that follows, this line must also represent a fold, and this line must also represent a fold. Actually I should tell you quickly, that in the world of trihedral objects. Y junctions can have multiple kind of constraints. But right now, let's just look at this one single constraint. So in the case of an L junction, which has a shape like this, in the world of trihedral objects, an L junction is characterized by this being a blade, and this being a blade, where a blade is a line, well we cannot infer that two surfaces are getting connected with each other. Again, the L-Constraint can actually have many more formulations. But right now, we're keeping it simple just looking at one single constraint for the

L junction. Similarly, in the world of trihedral objects, one of the ways in which a double junction gets characterized is through a blade, fold, blade. In effect, we're defining a spatial grammar here, for the world of trihedral objects. The equivalent of this, in case of grammar of natural language sentences might be that a sentence can have a non phrase, followed by their verb phrase, followed by a propositional phrase, and so on. Given this set of very simple constraints for the world of trihedral objects, let us see how these constraints actually can be propagated, to group edges and to surfaces

10 - Exercise Assembling the Cube I

[Click here to watch the video](#)

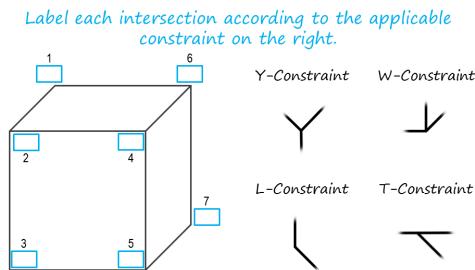


Figure 742: Exercise Assembling the Cube I

Let us do an exercise together. Here is a cube with its seven junctions. For each of these junctions, identify the kind of junction that it is.

11 - Exercise Assembling the Cube I

[Click here to watch the video](#)

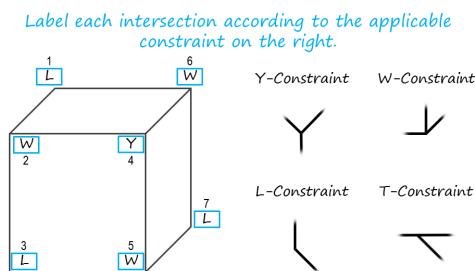


Figure 743: Exercise Assembling the Cube I

What did you write down, David? So going down the outside we have alternating L and W

junctions. An L in the bottom left, a W, an L and so on. Right in the center we have a Y. So we can differentiate a Y junction and a W junction, by saying that for a W junction one of the three angles has to be over 180 degrees. Whereas in a Y junction, all three are less than 180 degrees. A T junction happens when one of the angles is exactly 180 degrees. This sounds good. And now let us look at how we'll apply these constraints to identify the surfaces.

12 - Exercise Assembling the Cube II

[Click here to watch the video](#)

Label each line according to the constraint on the right.

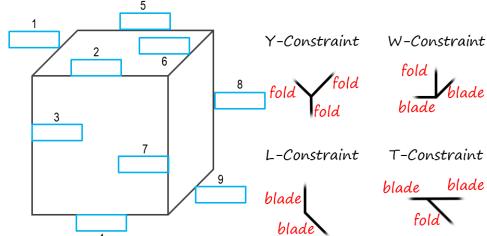


Figure 744: Exercise Assembling the Cube II

Let us do another exercise together. We have identified the type of each of these junctions. Let us now use, the constraints for each type of junction to identify the type of each of these edges. For each of these boxes, right? Either fold or blade for the type of the edge.

13 - Exercise Assembling the Cube II

[Click here to watch the video](#)

Label each line according to the constraint on the right.

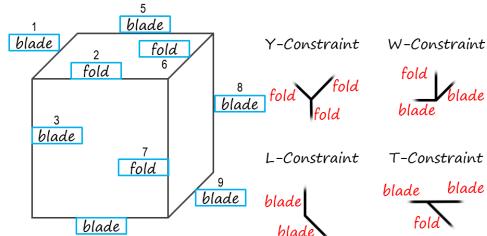


Figure 745: Exercise Assembling the Cube II

That's good, David. Note that David started on the top left corner, this is a random selection. He could have started at any other corner, for example, this one or that one. And found the same

answer and that is because we have simplified these constraints. But now that we know that this line is a fold, by the definition of fold, we know that two surfaces must be meeting at this line. It follows then, that this must be a surface and this must be a surface. Similarly, because we know this is a fold and by definition of a fold, two surfaces must be meeting here. Follows this is a surface, this is a surface and so on. And now we have identified that this one surface, this is another surface, this is a third surface. In this way, the visual system used knows the different kinds of junctions in the world of triangular objects, and it constrains at each of these junctions to figure out which of these lines made surfaces. Instead of thinking of this as one single surface, the visual system identified this as being composed of three different surfaces. And now we can recognize that this might be a cube.

14 - More Complex Images

[Click here to watch the video](#)

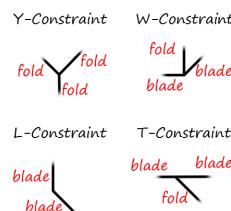
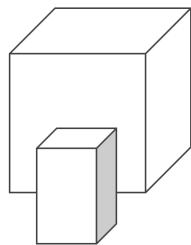


Figure 746: More Complex Images

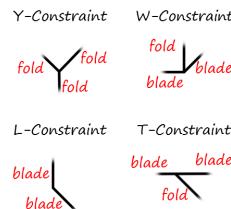
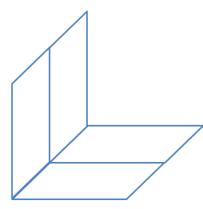


Figure 747: More Complex Images

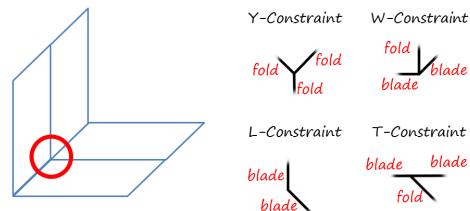


Figure 748: More Complex Images

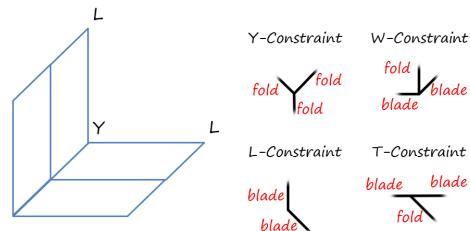


Figure 749: More Complex Images

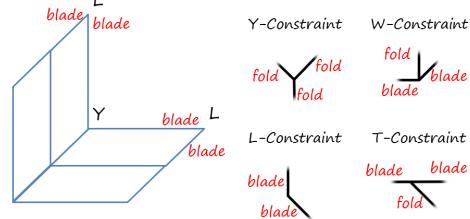


Figure 750: More Complex Images

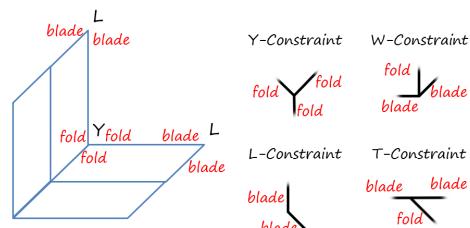


Figure 751: More Complex Images

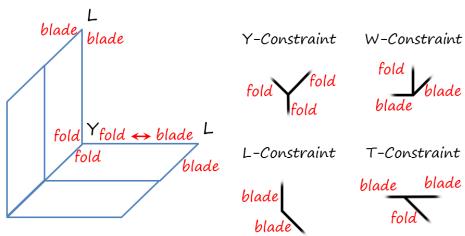


Figure 752: More Complex Images

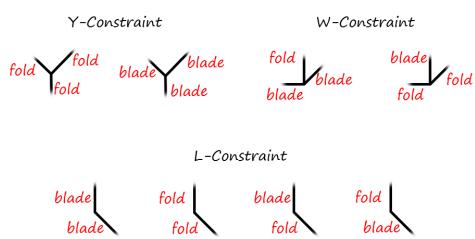


Figure 753: More Complex Images

Now of course, some of us do see this as a 3D shape. You can think of this as a paper folded here. One plane of the paper and another plane of the paper. This looks kind of like an open book. This particular line here then can be ignored, just being a line of these two planes, not signifying a surface by itself. If you view this only as a line, and not signifying a surface then it adresses David's first problem. But how do we address David's second problem of this being a fold or a blade depending upon where we started constraint propagation from? The answer is that we actually have a much more complex ontology of disconnections. The answer lies in the fact that we have so far used a very simple ontology, just to demonstrate the constraint propagation process. In reality the ontology risk constraints is more complicated. Let's do Y-constraint may not just fold, fold and fold, but it might also be blade, blade and blade. And the L-constraint is not always blade and blade and fold and fold. It could also be blade and fold and fold and blade. Now we can see David's second problem disappearing, because the Y junction may have a blade and the L junction may also suggest a blade. And there is then no conflict. Let

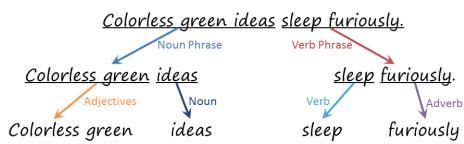
me know that what we have shown here, is still not a full anthology of the Y, W, L and T constraints. T constraints in particular, may have additional complexity. The advantage of having a more complete ontology is, that we can use that ontology to interpret more complex scenes like this one, where there are two rectangular objects, one being partially occluded by the other one. Of course, the more complicated ontology is not without its own problems. It now introduces ambiguities of a different kind. This particular junction. Is it now a blade, blade, blade, or is it a fold, fold, fold? Both of them are permissible in the new complete ontology. In order to resolve some of these ambiguities, we can come up with additional conventions. One convention is, that all of these edges that are next to the background, we'll consider them to be blades. So we'll make this a blade, blade, blade, blade. Once you make all these blades, then it's easy to propagate the constraints. Notice this W junction could have been a fold, blade, fold, or a blade, fold, blade. But if we adopt the convention of labeling all of these lines as blades, then, this W junction can only be blade, fold, blade. But if this is a fold, this Y junction can only be fold, fold, fold, and so on. And yet, helps us resolve the ambiguity of what this junction could be. This task of image interpretation is an instance of the abduction task. In abduction, we try to come up with the best explanation for the data. This is the data, we're trying to interpret it in terms of an explanation. We'll discuss abduction in more detail when we come to diagnosis. Well now notice that, we start with what we know. Blade, blade, blade. And that we propagate the constraints, so that we can disambiguate between other junctions.

15 - Constraint Propagation for Natural Language Processing
[Click here to watch the video](#)

Colorless green ideas sleep furiously.

Constraints:
 Sentence = Noun Phrase + Verb Phrase
 Noun Phrase = [Adjectives] + (Noun or Pronoun)
 Verb Phrase = Verb + [Adverb]

Figure 754: Constraint Propagation for Natural Language Processing



Constraints:
 Sentence = Noun Phrase + Verb Phrase
 Noun Phrase = [Adjectives] + (Noun or Pronoun)
 Verb Phrase = Verb + [Adverb]

Figure 755: Constraint Propagation for Natural Language Processing

Soft drinks due from thank bills insurance.

Constraints:
 Sentence = Noun Phrase + Verb Phrase
 Noun Phrase = [Adjectives] + (Noun or Pronoun)
 Verb Phrase = Verb + [Adverb]

Figure 756: Constraint Propagation for Natural Language Processing

Let us again write to the center. Colorless green ideas sleep furiously. You and I can quickly recognize, that this sentence is grammatically correct even as it is semantically meaningless. How do we do it? Consider this mini-grammar. This is a small subset of the English language grammar consisting of just three simple rules. The sentence can go into a noun phrase, followed by a verb phrase. A noun phrase can be, optional adjectives, followed by a noun or a pronoun. The square bracket here means, optional. A verb phrase, composed of a verb, followed by

optional adverb. The variables in this particular sentence, are the words. The values we assign to them are elliptical categories like verb, objective, noun and pronoun. If we can make a pastry for this, that assigned values to these variables in a way that is consistent with this grammar, then this particular sentence is grammatically correct. Let's try to make a pastry for this sentence. A sentence can be a noun phrase, or a word phrase. So we may say, that this is a noun phrase and this is word phrase. Of course at this particular point we do not know, where we should make this demarcation, what should go into the noun phrase, what should go into the word phrase? We know whether or not, this demarcation is correct depending upon whether or not this noun phrase, and word phrase meet the low level lexical categories. So let's look at colorless green ideas. We know that a noun phrase can go into one or more adjectives, followed by a noun or pronoun. So we can look at a lexicon, and know that ideas are a noun, so we say that this is a noun. We can look at a lexicon that tells us that colorless and green are adjectives, and so colorless and green are adjectives here. We have satisfied this part of the constraint. So I am ready for the verb phrase, a verb phrase can be composed of a verb, followed by one or more optional adverbs. We can look in the lexicon, and sleep is a verb, and furiously is an adverb, so we have satisfied the constraints for this particular part. Because we have satisfied the constraints, we know the top level demarcation of this as a noun phrase, and this as a verb phrase was correct. So the processing is not very top down here. It also has a bottom up component. In this way we are able to decide, that this particular sentence is grammatically correct because it satisfies the constraints of [our mini] grammar. Know that we have used knowledge of constraints, and the matter of constraint propagation, both for visual processing and for language processing. This sentence method is very general purpose in doing independent. Once we have done constraint propagation, to derive the parse tree for the sentence, then we can do additional processing. We can use this parse tree to support semantic analysis to build an understanding, a semantic un-

derstanding, of the sentence. Similarly, in visual processing, once we have used the constraints for doing line labelling. And recognize the surfaces and their orientations. We can then go on further and recognize the object in its 3D form.

16 - Assignment Constraint Propagation

[Click here to watch the video](#)

Assignment
How would you use constraint propagation to design an agent that could answer Raven's progressive matrices?

Figure 757: Assignment Constraint Propagation

So how would constraint propagation be useful for Raven's progressive matrices? This concept has a strong correspondence to the final project where you'll be asked to reason over the images of the problem directly instead of over the propositional representations that we've given you in the past. So first, if constraint propagation leverages a library of primitive constraints, what will your constraints be for the final project? How will you propagate those constraints into the image to understand it? Then once you've actually propagated those constraints, how will you actually use those inferences? Will you abstract out propositional representations? Or will you stick to the visual reasoning and transfer the results directly?

17 - Wrap Up

[Click here to watch the video](#)

To recap...

- Definition
- Image processing
- Natural language understanding
- Advanced problems

Figure 758: Wrap Up

So today we've talked about constraint propagation, which is a method of inference where we assign values to variables to satisfy certain external constraints. By doing so, we arrive at strong inferences about the problem, like which shapes represent objects, or how the words in a sentence interact. After defining constraint propagation, we talked about how it can be useful in interpreting images by using prior knowledge of constraints to anticipate predictable shapes out in the world. Then we talked about natural language understanding, where prior knowledge of the rules of grammar and parts of speech allow us to make sense of new sentences. Now, constraint propagation is actually an incredibly complex process. We have numerous constraints for visual scenes and verbal sentences that we haven't discussed here. We all see this constraint propagation in other areas as well, such as in making sense of auditory and tactile information. Reading braille for instance, can be seen as an instance of constraint propagation. We'll pick up on this discussion later when we talk about visual and spatial reasoning. But this will also come up in our discussion of configuration, which can be seen as a specific instance of constraint propagation in the context of design.

18 - The Cognitive Connection

[Click here to watch the video](#)

Constraint propagation also connects to human cognition. First, constraint propagation is a very general purpose method like, means analysis. In both knowledge based AI and in human cognition, constraint propagation allows us to use our knowledge of the world, in order to make sense of it. Constraints can be of any kind. Symbolic, as well as numeric. We discussed symbolic constraints in today's lesson. A good example of numeric constraints comes from XL spreadsheets, with which most of you are familiar. If the columns in a particular spreadsheet are connected to some formula, and you make a change in one column, then the change is propagated into all the columns of the spreadsheet. That's an example of Numerical Constraint Propagation. We have seen constraint propagation under

other topics as well. For example, planning, and understanding, and scripts. The next topic, configuration, will build on this notion of constraint propagation.

19 - Final Quiz

[Click here to watch the video](#)

All right. Please write down what you understood from this lesson, in the box right here.

20 - Final Quiz

[Click here to watch the video](#)

And thank you for doing it.

constraints. Constraint Propagation is used in many complex, real-life scenarios of Natural Language Understanding, Computer Vision/Image processing, etc.

References

1. Winston P., Artificial Intelligence, Chapter 12, Pages 249-266.

Optional Reading:

1. Winston Chapter 12, pages 249-266; [Click here](#)

Exercises

None.

Lesson 21 - Configuration



The initial configuration of the universe may have been chosen by God, or it may itself have been determined by the laws of science. In either case, it would seem that everything in the universe would then be determined by evolution according to the laws of science, so it is difficult to see how we can be masters of our own fate.

– Stephen Hawking.

01 - Preview

[Click here to watch the video](#)

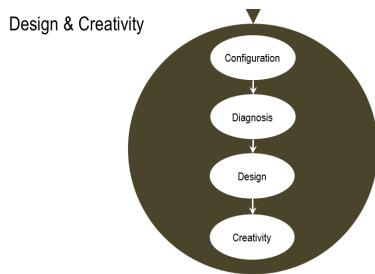


Figure 759: Preview

- Lesson Preview**
- Design & configuration
 - Plan refinement
 - Connections to earlier topics

Figure 760: Preview

Today, we'll talk about configuration. Configuration is a very routine kind of design task in which all the components of the design are already known. The task now is to assign values to

the variables of those components so they can be arranged according to some constraints. Configuration will be your first part under the unit on designing creativity. We'll start by talking about design, then we'll define configuration. Then we trace through the process of configuration, a specific measure called planned refinement. Finally, we'll connect configuration to several earlier topics we have discussed such as classification, case based reasoning, and planning.

02 - Define Design

[Click here to watch the video](#)

Let us talk about what is design. Design in journal takes us input some sort of needs, so goals or functions. It gives us output that's specification of this structure of some artifact that satisfies those needs and goals and functions. Note that the artifact need not be a physical product. It can be a process. A program, a policy. Some example for design, design a robot that can walk on water. Design a search engine that can return the most relevant answer to a query. The Federal Reserve Bank designs and monitor the policy to optimize the economy. Note the design is very wide ranging, open ended and ill-defined. In problem solving, typically the problem remains fixed, even as the solution evolves. In design, both the problem and

the solution co-evolve. The problem evolves as the solution evolves. We are studying design and AI because we are working with AI agents that can do design. At least potentially, we want AI agents that can design other AI agents.

03 - Exercise Designing a Basement

[Click here to watch the video](#)

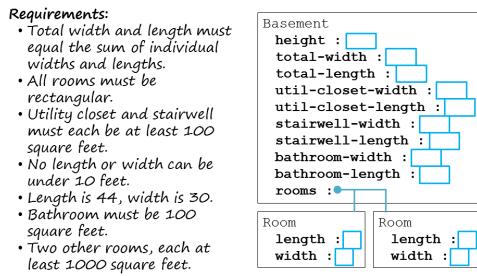


Figure 761: Exercise Designing a Basement

Thanks, Isuke so right now, my wife and I are actually building a house and as part of that, we need to configure the basement for the house. I've taken a list of some of the requirements for this basement and listed them over here on the left. And on the right, I have the variables that we need to assign values to, we have things like the width of the utility closet, the length of the stairwell, we also had two additional rooms, each must have their own length and width. So try to configure our basement, such that we meet all the different requirements listed over here on the left, write a number in each of these blanks.

04 - Exercise Designing a Basement

[Click here to watch the video](#)

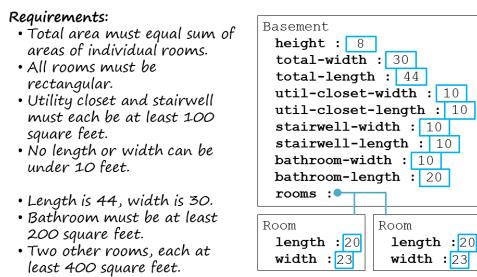


Figure 762: Exercise Designing a Basement

So I started off this process by assigning some variables I already knew. So the height being eight is just kind of a default value for the height of a basement. It could've been seven, it could've been nine. Anything that's tall enough that we can fit in it is fine. For the total width and the total length we were actually given those over here in the requirements. The length is 44 and the width is 30. Those are set by the other floors of the house, and the basement just kind of mimics their footprint. So, what I did next, was that I just went ahead and took the requirements for the utility closet and the stairwell and went ahead and applied them. I just said that the utility closet and the stairwell must each be 100 square feet. Let's just keep these simple and go ahead and assign those 10 by 10 for each. Similarly for the bathroom, the bathroom must be at least 200 square feet. And no length or width can be under 10 feet. Let's just make it 10 by 20, that's pretty easy. After assigning those three rooms, we had 920 square feet left, so I decided to keep things simple here as well, split it in two, two rooms that are each 460 square feet. And one way to do that would be to make them each 20 by 23. So some of you might notice that these rooms don't actually map to the 44 by 30 basement, because while the areas of these rooms add up to the same number of square feet, they aren't configurable into the right arrangement. For the purposes of this example we'll be using numbers. But, a real configuration exercise for this would also involve arranging the rooms. So, not only do they add to the same number of square feet, but they can also fit inside the same rectangle. Thank you, David. There are several things to note from David's analysis. One is that in configuration final we want an arrangement of all the components of all the parts. So in this case, finally we want an arrangement of all the rooms and these stairwells and these utility closets and so on. Not just the size of each one of them, but the actual spatial layout. Second, they would begin by assigning values to some variables here because he thought that this variables were more restricted. One can use a number of different heuristics for ordering the variables. Perhaps we can choose those variables which are most re-

stricted first. Or we can choose those variables that restrict others the most first. We can choose the most important variables first. The point is there can be a large number of variables and we can impose an ordering on them.

05 - Defining Configuration

[Click here to watch the video](#)

Configuration: A problem-solving activity that assigns values to variables to satisfy constraints.

Figure 763: Defining Configuration

Configuration is a kind of design that assigns values to variables to satisfy certain constraints. Design in general can be very complex and sometimes very creative. Configuration is a kind of very routine design. The kind of everyday mundane design, which is very very common. While in design in the beginning, you do not know the components or the arrangement of components looks like. In configuration, you already know all the components. You simply have to figure out an arrangement that will satisfy constraints. As an example, consider the furniture in the room in which you are sitting right now. All the components, all the pieces of furniture are already known. The room is fixed. You can't change the room either. But you certainly can move the furniture around. There might be some constraints. For example, if the room has a couch and a TV, then you might impose a constraint that a couch has to be in front of the TV and a certain number of feet away from it. Given the constraint that all the furniture must somehow fit into the room as well as the constraint that the TV must be in front of the couch, you might first decide to place the TV and the couch at specific locations. And then having decided on the location of the TV and the couch, decide on the location of the other pieces of furniture.

David, given the configuration is so common, can you think of other examples? So, it seems like we could talk a lot about the physical arrangement of things in different types of rooms. For example, when an airline configures a new airplane they have to configure the arrangement of the seats, the arrangement of the window, the arrangement of the emergency exits and various things like that. But I think configuration really goes beyond just configuring the physical layout of items. So, when I'm taking a picture with my camera phone, for example, I have to configure different options like the focus, whether or not the flash is on, my distance to what I'm taking a picture of, and a few other different things like that. In doing so I'm still balancing different restrictions. So, for example, if I'm taking a picture of a landmark I can only be so close to it. And therefore I might need to make different decisions based on the focus or the flash in order afford for how far I can be from the landmark. Design in general is a very common information processing activity. And configuration is the most common type of design. Now that we have looked at the definition of a configuration task, we're going to look at methods for addressing that task. Once again recall that the components in case of configuration are already known. We are deciding on the arrangement of those components. We are assigning values to specific variables of those components, for example, sizes.

06 - The Configuration Process

[Click here to watch the video](#)

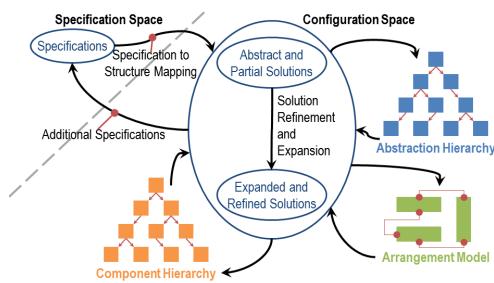


Figure 764: The Configuration Process

Here's an abstract specification of a

knowledge-based AI method for doing configuration design. The process starts with some specifications. These might be the specifications of all the constraints on this configuration problem, for example the constraints on David's basement. The output is an arrangement model, a model of the arrangement of all the components with the components already known, for example, the arrangement of David's basement. In this matter, for some configuration design, we begin with some very abstract, and perhaps partial solutions. This abstract and partial solution may be represented in the form of a design plan. So, an abstract and journal plan for basements of houses. Each plan specifies a subset of all the variables. We assign values to the variables in that plan. The plan is now complete. A completed plan can now be refined and expanded. At the next lower level, the plan specifies more variables. We assign values to those variables. Now that plan at the next level is complete, and we continue this iterative process until we have a complete arrangement model. As an example, if somebody was building a residential home, like David is doing right now, then the abstract plan may deal with the number of stories that the house will have. Once we assign to the variable number of stories, then we get a more expanded and refined plan where we have a plan for each of the stories including the basement. For example, for the main floor, there might be a plan which specifies something about a kitchen area, something about a living area, something about a bedroom area. As we assign values to the variables that define the living area, and the kitchen area, and the bedroom area, this plan gets completed. And now we might refine it further, a more detailed plan for the living area, for example. This abstraction hierarchy is a diagrammatic representation of this plan's arrangement from those abstraction. I begin with the most abstract plan, then I refine them and expand them as I go down. As I said earlier in configuration design, all the components are already known. Nevertheless, in some cases it might be able to also select the components. So for example, I might not only be able to reconfigure the TV, I might also be able to pick a specific kind

of TV. So this might be a TV in general and a more specific kind of TV and more specific kind of TV and so on. Note that the arrows coming here are two way arrows. This is intentional. In particular, let us look at the two way arrows between the arrangement model and the process of configuration and the specifications and the configuration process. Once this process has yielded an arrangement model, then we can assess the arrangement model and if needed go back to the process. As an example, the configuration process says that the TV is 12 feet away from the couch. And we assess it, and we decide the couch is too close to the TV. Then we can go back to the process and say, make the couch more than 12 feet away from the TV. That then becomes an additional specification here. And this is the meaning of the two way arrow. It is not just that we start with specifications and the configuration process works to satisfy them, but also, as the configuration process works and results in solutions, we can evaluate those solutions. And the specifications may change. This is a very common property of all design. One of the major differences between design and problem solving is that in problem solving, the problem typically remains fixed when we come up with a solution. In contrast in design, the problem evolves as the solution evolves. The problem and the solution co-evolve. So we start with a problem of satisfying certain constraints, but as the solution evolves, the evolution of the solution results in the evolution of a problem. As an example, let us suppose that the configuration problem is to configure the parts of a computer processor so that the processor can work at a particular speed. To do so, you come up with an arrangement model, but when you evaluate it, you find that the processor overheats. In that case, you may change the specification and say that the specification is not only that the processor should be fast enough, but also that it should not overheat. This is an example of problem evolution and solution evolution. Now that the problem has evolved, you may come up with a new solution for the processor design. This now is an example of problem evolution and solution evolution. So for an example that

might hit a little bit closer to home for many of you, as you've been designing your agents that can solve the Raven's test, you've done a process somewhat like this. You started with some specifications, general specifications, that your agent must be able to solve as many problems on the Raven's test as possible. You then start with an abstract solution of just a general problem solving process that you may have then refined to be more specific about the particular transformations to look for or the particular problem solving methods to use. That got you to your final result. But when you ran your file result, you may have found something like it would work but it would take a very, very long time to run, weeks or months. So that then causes you to revise your specifications. You not only need an agent that can solve as many problems as possible, but you also need one that can solve it in minutes or seconds instead of weeks or months.

07 - Example Representing a Chair

[Click here to watch the video](#)

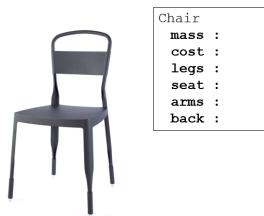


Figure 765: Example Representing a Chair

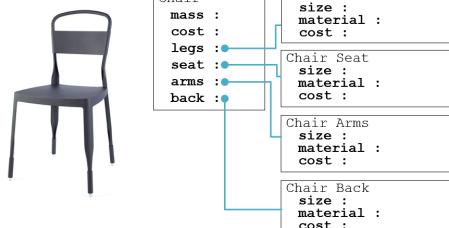


Figure 766: Example Representing a Chair

Let us look at how the process of configuration would work in detail. Recorder will met a

file of represent in reason, will represent knowledge and then reason over it. So suppose it asked the configure a chair. We must then somehow represent all of the knowledge of this chair then we can reason over it. So we will represent all of our knowledge of the chair in the form of a frame. Here's is the frame, here are the beta slots. For the time being, let's just assume that there are six slots that are important. The last four of the slots may point to their own frames. So there might be a frame for legs, a frame for seat, and so on. Now some of these frames have slots like size, and material, and cost. For legs, you may have an additional slot like count. What is the number of legs on the chair? And this particular frame representation captures of the knowledge of the generic, prototypical chair. To do configuration design is to come up with a specific arrangement of all the parts to this particular chair, to assign values to each of the variables, which means filling out the fillers for each of the slots. Of course, the values of these variables depends in part on the global constraints of mass and cost. We may, for example, have a global constraint on the mass of the chair, or in the cost of the chair, or both. We may have additional constraint as well. For example in material of a chair might become an, perhaps the material needs to be metal.

08 - Example Ranges of Values

[Click here to watch the video](#)

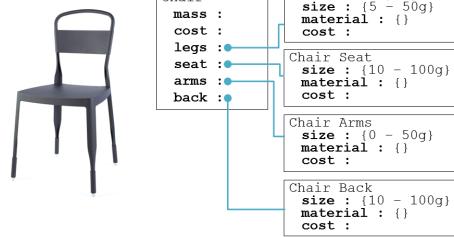


Figure 767: Example Ranges of Values

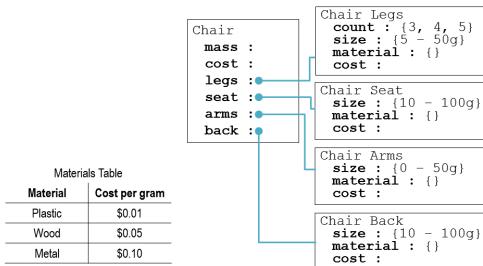


Figure 768: Example Ranges of Values

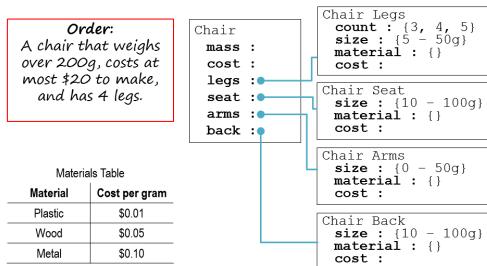


Figure 769: Example Applying a Constraint

Now in configuration design, we not only know all the components like legs, and set, and arms, and so on. We not only know the variables for each of the components, like size and material, and cost. But we also know the ranges of values that any of these variables can take. Thus the seat of a chair may have a certain weight, or length, or depth. Here between the sides and the seat in a very simple matter in terms of the mass of the seat as measured in grams. So 10 to 100 grams, you'll see in minute why we're using this simple measure. So when it is brackets for this material slot suggests that there is a range here, it will show the range on the left [in just a minute]. The cost then will be determined by the size and the materials. Let us suppose that this table captures the cost per gram for certain kinds of materials. Now you can see why we're using gram as a measure for the size of the seat. We wanted to very easily relate the size to the cost. The material slot now can take one of these three values. This is the range of values that can go into the material slot. Given a particular size and a particular material, we can calculate this cost. Note that this representation allows us to calculate the total mass of the chair and the total cost of the chair, given the total mass and the total cost at least of the components.

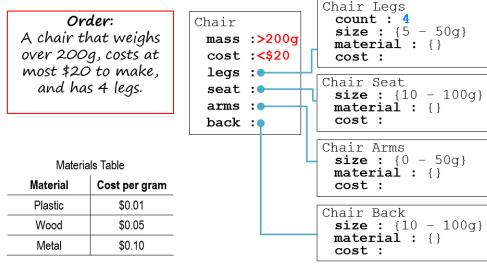


Figure 770: Example Applying a Constraint

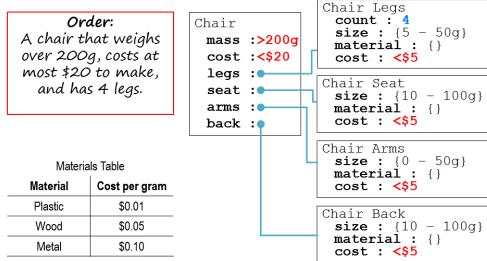


Figure 771: Example Applying a Constraint

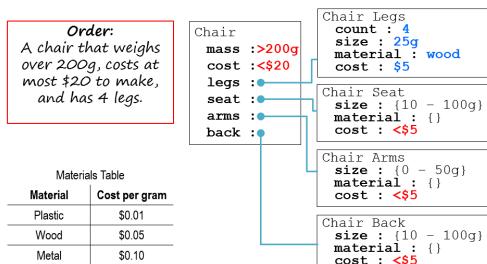


Figure 772: Example Applying a Constraint

09 - Example Applying a Constraint
Click here to watch the video!

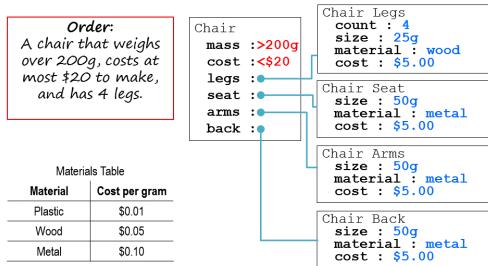


Figure 773: Example Applying a Constraint

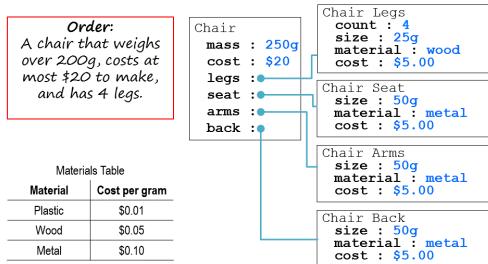


Figure 774: Example Applying a Constraint

Now let us suppose that we get a new order in which a customer wants a chair that weighs over 200 grams, costs at most \$20, and has four legs. Given the specification, what configuration process can use this knowledge to fill in the values of all the variables to satisfy the specification. So the first thing the process might do is to write down all the constraints that are given as part of the input specification. So the mass is greater than 200 grams, the cost is less than \$20, and the count of legs is 4. Now suppose that a configuration process has an abstract plan which first decides on the value of the cost variable before it decides on other variables. Let us also further suppose that this plan for deciding the cost evenly distributes the cost between the greatest components until unless specified otherwise by the specification. In this case the cost plan distributes this cost of \$20 between the four components and assigns less than five for each one of them. Now we define an expanding plan. This is two aspects to it, refine and expand. Index and aspect we deal with the components instead of the chair as a whole. And the define aspect we deal with more detailed variables that were not there in the chair. Consider the component legs,

for example. We already know the count, four, in the input specification. We know the cost, no more than \$5 from the higher level plan. Now we can design by using the other two variables, 25 grams and wood, for example. We can do the same for the other components. As we assign values to the variables of each of these components, we get a complete arrangement of all these components here, with values assigned to each of the variables. Given the specific values we assign to the variables for each of the components, we can now compute whether the constraint given in the input specification are satisfied. In this particular example, both the mass and the cost of the chair satisfy the input constraints. Note that the define and expands step in this particular process might have operated a little differently. It is also possible that define and expand step might say, the less, decide on the material before we decide on any of the other features. Plus within this complex configuration process, different designers may use different plans and different plans to find expansion mechanisms. Of course it is also possible that once we have a candidate solution, the candidate solution may not necessarily satisfy the input constraints. So the cost may turn out to be more than \$20, for example. In that case there are two options. Either we can iterate on the process, loading the cost, or we can go about changing the specification.

10 - Exercise Applying a Constraint

[Click here to watch the video](#)

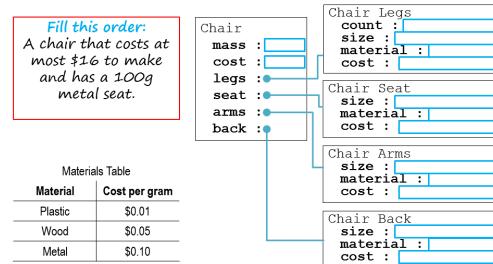


Figure 775: Exercise Applying a Constraint

Let us do an exercise together. This exercise again deals with the configuration of a chair. The input specification is a chair that costs at

most \$16 to make, and has 100 grams metal seat. Please fill out the values of all of these boxes. Try to use a configuration process that we just described, and make a note of the process that you actually did use

11 - Exercise Applying a Constraint

[Click here to watch the video](#)

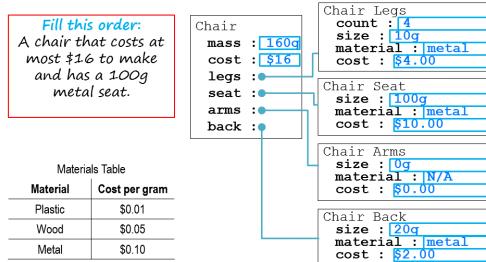


Figure 776: Exercise Applying a Constraint

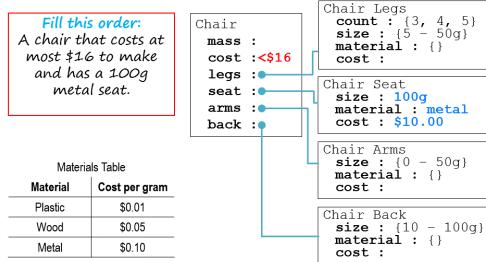


Figure 777: Exercise Applying a Constraint

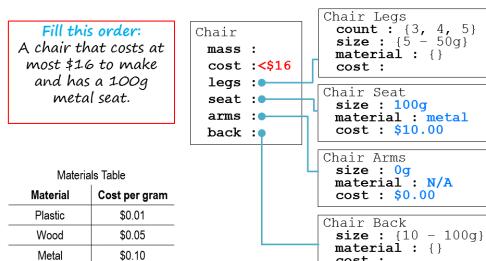


Figure 778: Exercise Applying a Constraint

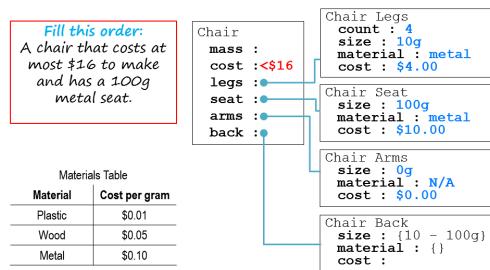


Figure 779: Exercise Applying a Constraint

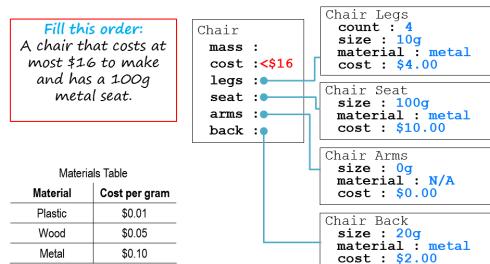


Figure 780: Exercise Applying a Constraint

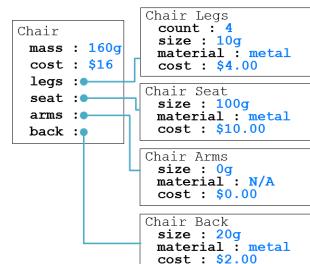


Figure 781: Exercise Applying a Constraint

So David, how did you configure the chair? So you can see here my final configuration for this chair. I ended up with a 160 gram chair. It cost \$16 based on having four metal legs. I didn't give it arms. I gave it a metal back. And of course the 100 gram metal seat that was required. As far as how I actually did this though, like before, I started off by writing out the initial constraints we were given. The chair has to cost less than \$16. It has to have 100 g metal seat. And it has 100 g metal seat. And that seat has to cost \$10. So 100 grams times \$0.10 per gram means \$10 for the seat. After this, however, we're already pretty close to our price limit. So the plan I'm using might specify that when

we're getting within a certain range of our price limit to operate under a heuristic that calls for minimizing the cost. That heuristic in that plan might then say, find the part that can be minimized the most next. That part would be the arms. The arms had a range from 0 grams to 50 grams, meaning that arms are not even required in our chair. So to minimize our costs, we're going to go ahead and cut the arms, and say that our arms cost nothing. Now our plan might recognize that we're no longer quite as constrained by our price, so we're going to choose the next most important part of the chair, which might be the legs. Given some information that the plan has about the ideal legs, it might choose to have four legs, that are ten grams each and are made of metal, which gives us a cost of \$4. Based on the \$10 seat and the four dollar legs, we now know that we have \$2 left for our back. The plan may have a heuristic that says it's ideal to match materials or the plan may have a heuristic that says that metal is the optimal material to use. So it may choose metal. And given \$2 for the remaining money and a material of metal, it can derive that 20g is the mass available for the back. And now that all of these individual variables have been assigned, we can then see that the final mass is 160g, and has a cost of \$16. At most \$16 meets that constraint. And we have our 100g metal seat over here. That's good David. This important note that David used several different kinds of knowledge. First, he had knowledge of the general chair. He knew about the components, he knew about the slots, but not necessarily all the fillers for the slots. Second, he had heuristic knowledge, he used the term heuristic. Recall that heuristic stands for rule of thumb. So heuristic knowledge about how to go about filling the values of some of these slots. Third, explicit list is not just knowledge about legs and seats and arms and so on, but also how this chair, as a whole, is decomposed into its components. That is one of the fundamental rules of knowledge and knowledge-based AI. It allows us to structure the problem so the problem can be addressed efficiently. Note that this process of configuration design is closely related to the method of constrained proposition

that we discussed in a previous lesson. Here are some constraints, and these constraints have been propagated downwards in the plan abstraction hierarchy.

12 - Connection to Classification

[Click here to watch the video](#)

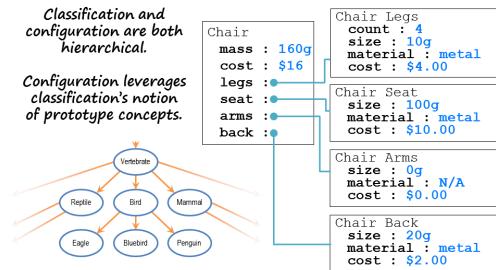


Figure 782: Connection to Classification

Configuration is also connected to classification. In case of classification, you establish and then define, establish and then define. In case of configuration, you're going to extend the plan, assign the values to the variables, refine and expand. Extend the plan, assign values to the variables, refine and expand. Second configuration leverages classification's notion of prototypical concepts. So the hierarchy of plans is organized around these prototypical notions of components of a chair. A chair typically consists of legs, and seats, and arms, and back, and so on. The ranges of values that these variables can take are also part of a prototypical knowledge of various components of a chair. In fact they are part of the default value of this particular component. Clearly there are also differences between classification and configuration. We use classification to make sense of the world by mapping combinations of precepts into equivalence classes. We use configuration to act on the world by designing actions. So it sounds to me like while classification is a way of making sense of the world, configuration is a way of creating the world. With classification, we perceive certain details in the world and decide what they are. With configuration, we're given something to create and we decide on those individual variables.

13 - Contrast with Case-Based Reasoning

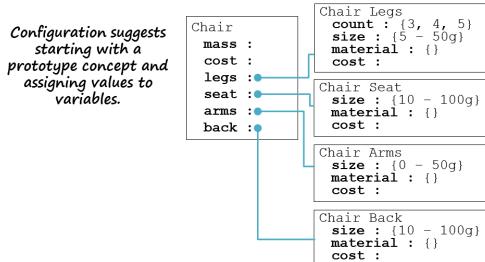
[Click here to watch the video](#)

Figure 783: Contrast with Case-Based Reasoning

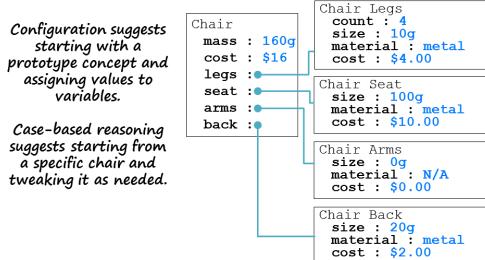


Figure 784: Contrast with Case-Based Reasoning

We can also cross configuration with case based reasoning. Both configuration and case based reasoning are typically applied to routine design problems, problems of the kind that we've often encountered in the past. Given a configuration, we start with a prototypical concept, then assign values to all the variables as we saw in this chair example. In case of case-based reasoning we start with the design of a specific chair that we had designed earlier. Look at its variables and tweak it as needed to satisfy this constraint so the current problem. Case-based reasoning assumes that we already designed our other chairs, and we have stored examples of the chairs for designing the memory. Configuration assumes, that we already designed enough chairs so that we can in fact extract the plan. When a specific problem is presented to an IA agent, the IA agent, if it is going to use the method of configuration, is going to call upon the plan obstruction

hierarchy and then start defining plans. If the AI agent uses the method of case based reasoning, then it'll go into the case memory, retrieve the closest matching case, and then start tweaking the case. Little bit later we will see how an AI agent select between different methods that were able to address the task. As we have mentioned earlier in the course, the chemical periodic table was one of the really important scientific discoveries. Similar to chemical periodic table, we are trying to build a periodic table of intelligence. Unlike the chemical periodic table which deals with balance electrons. Our periodic table of intelligence, deals with tasks and methods. In this particular course, we have considered both a large number of tasks, configuration being one of them, as well as a large number of methods, [plan] instantiation and case-based reasoning being two of them.

14 - Connection to Planning

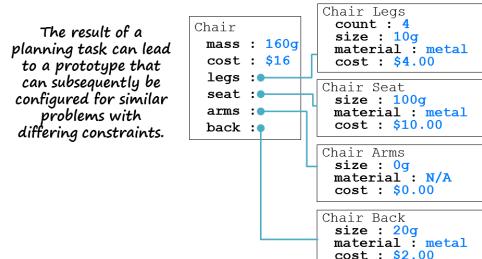
[Click here to watch the video](#)

Figure 785: Connection to Planning

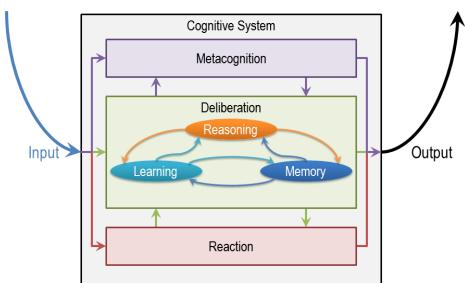


Figure 786: Connection to Planning

The process of configuration is also related to planning. You can consider a planner that actually generates the plan in this plan obstruction

hierarchy. But then for any plan in this plan obstruction hierarchy, then it converts a plan in this plan obstruction hierarchy into a skeletal plan. It drops the values of the variables in the plans and constructs it into a plan it's simply specify the variable without specifying the values. The process of configuration planning then, takes these plans, organizes them into obstruction hierarchy and goes about [instantiating] shading and refining and expanding them. We already discussed how configuration is connected to a number of other lessons like case based reasoning, planning and classification. You may also consider this plan to be kind of strict for physical object. In addition, this plans have been learned, through learning methods similar to the method of incremental concept learning. In addition, this plan hierarchy might be learned through learning methods similar to the method for incremental concept learning. One of the things that we are doing in knowledge based AI is, to describe the kinds of knowledge that we need to learn. Before we decide on what is a good learning method, we need to decide on what is it we need to learn? The configuration process tells us of the different kinds of knowledge that then become targets of learning. To connect this lesson back to our cognitive architecture, consider this figure once again. So knowledge of the prototypical chair, as well as knowledge about the radius, plans, and the abstraction hierarchy are stored in memory. As the input gives specification with the design problem, the reasoning component instantiates those plans, refines them and expands them. The knowledge itself is learned through examples of configuration of chairs that presumably, the agent is already encountered previously.

Assignment

How would you use configuration to design an agent that could answer Raven's progressive matrices?

Figure 787: Assignment Configuration

So how might you use the idea of configuration to design an agent that can answer Raven's progressive matrices? We've talked in the past about how constraint propagation can help us solving these problems. If configuration is a type of constraint propagation, how can you leverage the idea of variables and values in designing your agent? What are the variables and what values can they take? We've also discussed how planning can be applied to Raven's progressive matrices. If configuration leverages old plans, how you build your agent to remember those old plans and reconfigure them for new problems? Will it develop the old plans based on existing problems, or will you hand it the problems in advance?

16 - Wrap Up

[Click here to watch the video](#)

To recap...

- Design
- Defining configuration
- Process of configuration
- Connections to earlier topics

Figure 788: Wrap Up

So today we've talked about configuration, a kind of routine design task. We do configuration when we're dealing with a plan that we've used a lot in the past, we need to modify to deal with some specific new constraints. So for example, we've built thousands of buildings, and thousands of cars, and thousands of computers,

15 - Assignment Configuration

[Click here to watch the video](#)

and each of them is largely the same. But there's certain parameters, like the number of floors in a building, or the portability of the computer, that differ from design to design. So we need to tweak individual variables to meet those new constraints. We started this off by defining design in general, and then we used that to define configuration, as a certain type of routine design task. We then discussed the process of configuration and how it's actually very similar to constraint propagation that we've talked about earlier. Then we connected this to earlier topics like classification, planning and case-based reasoning, and saw how in many ways, configuration is a task, while other things we've talked about provide us the method for accomplishing that task. So now we'll move on to diagnosis, which is another topic related to design, where we try to uncover the cause of a malfunction in something that we may have designed. In some ways, we'll see that diagnosis is a lot like configuration in reverse.

17 - The Cognitive Connection

[Click here to watch the video](#)

Design is a very common cognitive activity. Some people even claim that design is a single cognitive activity that has the most economic value of all such activities. Configuration is a type of routine design, that occurs every single day. For example, you need to run some errands. You know the roads, you know the vehicle, you know the traffic patterns. Now you need to configure the specific route that can optimize some constraint such as time. Cooking is another everyday example of configuration. We know the recipes, which tell us about the high level plans and the ingredients we need to assign values to specific variables that can optimize some constraints such as taste. Notice that we can separate task from method. Configuration is a task

that can be addressed by many methods. We will look at several of them, such as [Case-Based Reasoning]. Plant refinement, [Case-Based Reasoning] test, and so on.

18 - Final Quiz

[Click here to watch the video](#)

Please summarize what you learned in this lesson in this blue box.

19 - Final Quiz

[Click here to watch the video](#)

Great, thank you very much.

Summary

Configuration is a very routine kind of design task in which all the components of the design are already known. The task now is to assign values to the variables of those components so they can be arranged according to some constraints.

References

1. Stefk, M. Introduction to Knowledge Systems, Pages 608-621, 656-666.

Optional Reading:

1. Stefk, Chapter 8 Parts 1; T-Square Resources (Stefk Configuration Part 1_Pgs 608-621_.pdf)
2. Stefk, Chapter 8, Part 2; T-Square Resources (Stefk Configuration Part 2_Pgs 656-666_.pdf)

Exercises

None.

Lesson 22 - Diagnosis



When you have eliminated the impossible, whatever remains, no matter how improbable, must be the truth.

— Sir Arthur Conan Doyle, *The Sign of Four*.

Listen to your patient, he is telling you the diagnosis.

— William Osler.

01 - Preview

[Click here to watch the video](#)

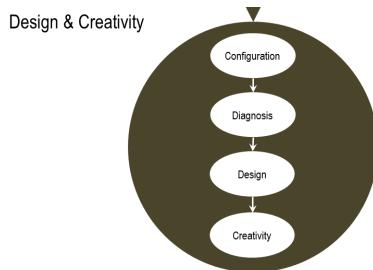


Figure 789: Preview

system it could be a car, a computer program, an organism, or the economy. Diagnosis builds on our discussion of classification and configuration. They start by defining diagnosis. They really setup two spaces. A data spaces and a hypothesis space. Data about the malfunctioning system. Hypothesis about the fault that can explain that malfunctioning system. Then, we'll constructing mappings through data space to hypothesis space which amount to diagnosis. We'll discuss two views of diagnosis, diagnosis as classification and diagnosis as abduction. Abduction in this context is a new term to you. We'll discuss it in more detail today.

Lesson Preview

- Defining diagnosis
- Data and hypothesis spaces
- Mapping data to hypotheses
- Two views of diagnosis

02 - Exercise Diagnosing Illness

[Click here to watch the video](#)

Patient:

A: Normal
B: High
C: Low
D: Normal
E: Normal
F: Normal
G: Normal
H: Low

Illnesses:

- Alphitis: Elevated A, Reduced C, Elevated F
- Betatosis: Elevated B, Reduced C, Elevated E, Reduced H
- Gammaonoma: Elevated D, Elevated E, Elevated F
- Deltaclot: Elevated B, Reduced C
- Episcusus: Reduced H
- Zetad: Elevated B, Reduced C, Reduced E, Reduced F
- Etaemia: Elevated A, Reduced D, Reduced H
- Thetadesis: Elevated B, Reduced C, Reduced H
- Iotalgia: Elevated A, Reduced E, Elevated F, Elevated G
- Kappacide: Reduced A, Reduced F, Reduced G
- Lambdacite: Reduced A, Reduced E, Reduced F, Reduced G
- Mutension: Elevated A, Elevated G

What illness (or set of illnesses) would you use to diagnose this patient?

Figure 790: Preview

Today we will talk about diagnosis. Diagnosis is the identification of the fault or faults responsible for a malfunctioning system. The

Figure 791: Exercise Diagnosing Illness

To illustrate the task of diagnosis, let us begin with an exercise. When we think of diagnosis, most of us think in terms of medical diagnosis. The kind of diagnosis a doctor does. So this particular exercise, is coming from medical diagnosis, actually it's a made-up exercise from medical diagnosis. And here's a set of diseases, fictional diseases, that the doctor knows about, along with the symptoms that each disease causes. So, Alphaitis, for example, causes elevated A, reduced C and elevated F and so on. Given this set of data, and this set of diseases, what disease or set of diseases do you think the patient suffers from?

03 - Exercise Diagnosing Illness

[Click here to watch the video](#)

Patient:	Illnesses:
A: Normal	<input type="radio"/> Alphaitis: Elevated A, Reduced C, Elevated F
B: High	<input type="radio"/> Betatosis: Elevated B, Reduced C, Elevated E, Reduced H
C: Low	<input type="radio"/> Gammaonitis: Elevated D, Elevated E, Elevated F
D: Normal	<input type="radio"/> Deltafoc: Elevated B, Reduced C
E: Normal	<input type="radio"/> Epsicusus: Reduced H
F: Normal	<input type="radio"/> Zetad: Elevated B, Reduced C, Reduced E, Reduced F
G: Normal	<input type="radio"/> Etæmia: Elevated A, Reduced D, Reduced H
H: Low	<input checked="" type="radio"/> Thetaidesis: Elevated B, Reduced C, Reduced H <input type="radio"/> Iotalgia: Elevated A, Reduced E, Elevated F, Elevated G <input type="radio"/> Kappacide: Reduced A, Reduced F, Reduced G <input type="radio"/> Lambdacrite: Reduced A, Reduced E, Reduced F, Reduced G <input type="radio"/> Mutension: Elevated A, Elevated G
What illness (or set of illnesses) would you use to diagnose this patient?	

Figure 792: Exercise Diagnosing Illness

That's a good answer, David. Note that David did several things in coming up with his answer. First, he made sure that his answer covers all these signs and symptoms. This is the principle of coverage. We want to make sure that the diagnostic conclusion actually accounts for all the input data. Second, we chose a single hypothesis over a combination of hypothesis, although the combination could have explained this data as well. This is the principle of parsimony. In general, we want a simple hypothesis for explaining the entire data. Third, these hypotheses can have greatest interactions between them, and these interactions can make your diagnostic task quite complicated. Fourth, they would use the term explanation. This is an important aspect for diagnosis. We want a set of hypotheses that could explain the input data. Now this [turns out to be a relatively simple exercise] we

did it in this simple exercise, because there is one single disease that can, in fact, explain all the input data. What would happen if there was no single hypothesis that could cover the entire input data? Or what would happen if there were multiple hypotheses that could equally well explain the input data? [turns out to be a relatively simple exercise] diagnostic task can be quite complicated.

04 - Defining Diagnosis

[Click here to watch the video](#)

Diagnosis: To determine what is wrong with a malfunctioning device.

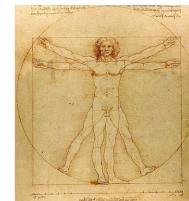


Figure 793: Defining Diagnosis

Diagnosis: To determine what is wrong with a malfunctioning device.



Figure 794: Defining Diagnosis

We may define the task of diagnosis as determining what is wrong with a malfunctioning device. Or more generally, what is the fault that is responsible for a malfunctioning system. Given a system, we expect some behavior from it. We expect it to do something. However, we may observe that the system is doing something different. So there is the expected behavior and there is the observed behavior, and there is a discrepancy between them. When there is a discrepancy, we know that the system is malfunctioning. The question then becomes what is the fault or the set of faults responsible for the malfunctioning system. When we think of diagnosis

we typically think of medical diagnosis. But, of course diagnosis can occur in a very large number of domains. Here are three diagnostic domains with which all of us are familiar. The first figure shows the engine of a car. When I insert the key into the ignition system, I expect the engine to turn on. That is the expected behavior. But suppose that I insert the key, and the engine doesn't turn on. That's the observed behavior. There is a discrepancy between the expected behavior and the observed behavior. When I insert the key into the ignition, I expect the engine to turn on. That's the expected behavior. But then suppose that I put the key in, and the engine doesn't turn on. That's the observed behavior. The discrepancy between the expected behavior, turn the engine on, and the observed behavior, the engine doesn't turn on, so I know there is a malfunction. Given this malfunction, the question becomes what is at fault, the force responsible for it and that's the diagnostic task. To address this diagnostic task, I may use a rule which says that if the engine doesn't turn on when the key is inserted, check the carburetor. Suppose I go and check the carburetor and everything is okay with the carburetor. Then I [Thus, the] get activated. And let me say, if the engine doesn't turn on and everything is okay with the carburetor, go check the spark plugs. And this way, I am going to use a production system to isolate the fault of faults responsible for the malfunction. Something similar happens with computer hardware repair. When we turn on a computer, there are few behaviors that we expect of it. We expect it to boot up quickly, we expect it to run fast, and we expect it to stay cool. Now, imagine we turn a computer and notice it started to run at a much higher temperature than we're accustomed to. We might remember that the last time we encountered this problem, there were problems with the fan. So we might use that to then diagnose this a problem with the fan and replace the fan. This happens at the software level too. If I'm writing a program and the output differs from what I was expecting, I set about debugging the program and finding the fault. One way of doing that is called a rubber duck debugging, which involves explaining

my model of how my program works to the rubber duck so that I might uncover the error by forcing myself through an explanation process. Note that we discussed the same diagnostic task in three different domains. In each domain there was a discrepancy between the expected and the observed behaviors and we tried to identify the fault, or faults responsible for it. Note also that we alluded to three different methods for doing diagnosis. The matter of rule based reasoning, the matter of case based reasoning, and the matter of model based reasoning. We haven't talked a lot about the matter of model based reasoning so far. We will do so when we come to systems thinking later in the class. Of course we can use the matter of rule based reasoning, not only for diagnosing car engines but also for repairing computer hardware or for diagnosing computer software. In this particular lesson our focus will be on the diagnostic task. By now all of us are already familiar with many reasoning methods that are potentially applicable to class.

05 - Data Space and Hypothesis Space

[Click here to watch the video](#)

Data Space	Hypothesis Space
D ₁	H ₁
D ₂	H ₂
D ₃	H ₃
D ₄	H ₄
D ₅	H ₅
D ₆	H ₆
D ₇	H ₇
D ₈	H ₈
D ₉	H ₉
D ₁₀	H ₁₀
D ₁₁	H ₁₁
D ₁₂	H ₁₂
D _N	H _N

Figure 795: Data Space and Hypothesis Space



Data Space	Hypothesis Space
D ₁	H ₁
D ₂	H ₂
D ₃	H ₃
D ₄	H ₄
D ₅	H ₅
D ₆	H ₆
D ₇	H ₇
D ₈	H ₈
D ₉	H ₉
D ₁₀	H ₁₀
D ₁₁	H ₁₁
D ₁₂	H ₁₂
D _N	H _N

Figure 796: Data Space and Hypothesis Space

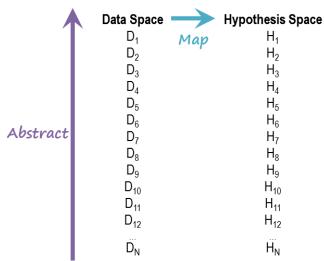


Figure 797: Data Space and Hypothesis Space

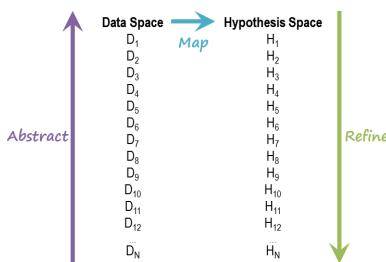


Figure 798: Data Space and Hypothesis Space

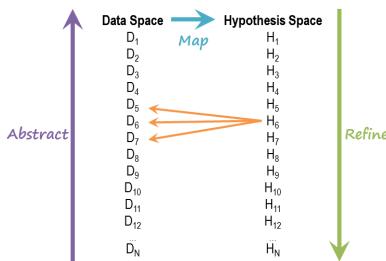


Figure 799: Data Space and Hypothesis Space

We can think of diagnosis as a mapping from a data space, to a hypothesis space. In case of a medical diagnosis, the data may be the greatest kind of signs and symptoms that I may go to a doctor with. Some of the data may be very specific, some of it may be very abstract, an example of a very specific data is that a [another rule] temperature is 104 degrees fahrenheit. An example of the extraction of the data is that [another rule] is running a fever. The hypothesis space consists of all hypothesis that can explain parts of the observed data. A hypothesis in the hypothesis space can explain some part of the data. In case of medicine, this hypothesis may reference to diseases. A doctor may say that

my hypothesis is that a shook is suffering from flu, and that explains his high fever. In the domain of car repairs, this hypothesis may refer to specific faults with the car, for example, the carburetor is not working properly. In the domain of computer software, this hypothesis may refer to specific methods not working properly. And this mapping from data space to the hypothesis space can be very complex. The complexity arises partly because of the size of data space, partly because of the size of hypothesis space, partly because the mapping can be M to N. And also, because this hypothesis can interact with each other, If H3 is present, H4 may be excluded, If H5 is present, H6 is sure to be present and so on. It helps then not to deal with all the raw data, but to deal with abstractions of the data, so the initial data that a patient may go to a doctor with may be very, very specific. The signs and symptoms of their particular specific patient, but the diagnostic process might abstract them from Asok has a fever of 104 degrees fahrenheit to Asok has a high fever. This abstract data that can be mapped into an abstract hypothesis, Asok has high fever can get mapped into Asok has a bladder infection for example. The abstract hypothesis can now be refined into a suffering from flu or a flu for a particular screen. At the end, we want a hypothesis that is as refined as possible, and that explains all the available data. When we were talking about classification, we talked about two processes of classification, bottom-up process and our top down process. The bottom up process of classification, we started with raw data and then grouped and abstracted, it in case of top down classification we started with some high level class and then established it and refined it. You can see that in diagnosis both the bottom up process of classification, and the top down process of classification are co-occurring. This method of bottom up classification and data space, mapping and hypothesis space, and then top down classification of hypothesis space is called heuristic classification. This is yet another method like rule-based reasoning, case-based reasoning, and model-based reasoning with a diagnostic task.

06 - Problems with Diagnosis as Classification

[Click here to watch the video](#)

Problem #1: One data point, multiple hypotheses.

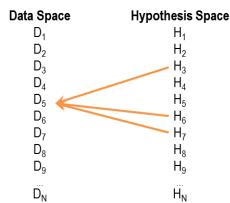


Figure 800: Problems with Diagnosis as Classification

Problem #2: One hypothesis, multiple sets of data.

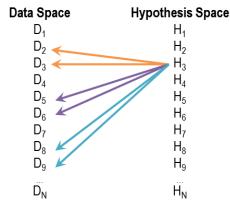


Figure 801: Problems with Diagnosis as Classification

Problem #3: Multiple hypotheses, multiple sets of data.

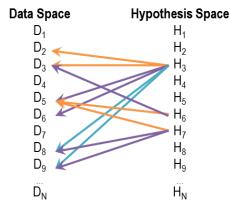


Figure 802: Problems with Diagnosis as Classification

Problem #4: Mutually exclusive hypotheses.

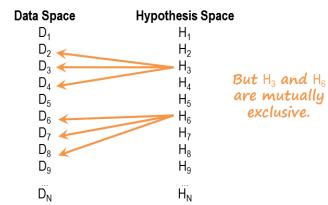


Figure 803: Problems with Diagnosis as Classification

Problem #5: Interacting data points.

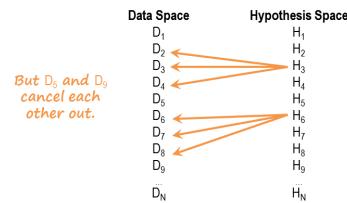


Figure 804: Problems with Diagnosis as Classification

Patient:	Illnesses:
A: Normal	<input type="radio"/> Alphaitis: Elevated A, Reduced C, Elevated F
B: High	<input type="radio"/> Betatosis: Elevated B, Reduced C, Elevated E, Reduced H
C: Low	<input type="radio"/> Gammaonoma: Elevated D, Elevated E, Elevated F
D: Normal	<input type="radio"/> Deltacol: Elevated B, Reduced C
E: Normal	<input type="radio"/> Episcusus: Reduced H
F: Normal	<input type="radio"/> Zetaf: Elevated B, Reduced C, Reduced E, Reduced F
G: Normal	<input type="radio"/> Etaemia: Elevated A, Reduced D, Reduced H
H: Low	<input checked="" type="radio"/> Thetadesis: Elevated B, Reduced C, Reduced H
What illness (or set of illnesses) would you use to diagnose this patient?	
<input type="radio"/> Iotalgia: Elevated A, Reduced E, Elevated F, Elevated G	
<input type="radio"/> Kappacide: Reduced A, Reduced F, Reduced G	
<input type="radio"/> Lambdacite: Reduced A, Reduced E, Reduced F, Reduced G	

Figure 805: Problems with Diagnosis as Classification

Several factors conspire to make this process of classification much more complicated. This is as you would expect in AI. If it was an easy problem it would not be part of AI. The first factor that makes this problem complicated is that one data point might be explained by multiple hypotheses. So I go to the doctor with high fever, D5 here, and several hypotheses about different diseases might explain my high fever. Which of these hypotheses, then, is true? A second factor that complicates things is that one hypotheses

may explain multiple sets of data. So the hypotheses that Ashok has influenza might explain not only that he has fever, but also that he feels tired, and also that he is shivering, and also that he can't sleep at night. Go to your doctor with two data items, one that have high fever and the other that I am tired. Now the doctor may come up with a hypothesis, H3, that Ashok suffers from flu. However, when H3 is present, then one can expect other symptoms to be observed as well. However, the hypothesis of H3 may generate expectations but additional data items. How, then, may a doctor decide if H3 is true? Well, one possibility is that a doctor may ask Ashok additional questions to collect additional data. Do you shiver at night, the doctor may ask, if that is one of the expectations generated by the hypothesis of having flu. because the mapping is not only from the data space to the hypothesis space, the mapping is also from the hypothesis space to the data space. Diagnosis entails not only mapping data to hypothesis, but also to know the expectations of additional data and, collecting that additional data. Of course, both of the first two factors may be present at the same time. That is, one hypothesis may explain multiple data items. And multiple hypothesis may explain the same data item. So, in general, this is a M to M mapping, multiple hypothesis, multiple sets of data. And, of course, this immediately makes the diagnostic task harder. The fourth factor that makes the diagnostic task hard is that these hypothesis could interact with each other. One of the common interactions between hypotheses is called mutual exclusion. Mutual exclusion occurs if one hypothesis present, another hypothesis cannot be true. In this case, H3 explains D2, D3, D4. And H6 expands D of 6, D of 7, D of 8. But if H3 is present, H6 cannot be true. And if H6 is present, H3 cannot be true. This makes the diagnostic task hard because if a patient goes to a doctor with symptoms D3, D4 and D6, D7, then the question becomes whether to include H3 or to include H6 to define our conclusion. A fifth factor that makes the diagnostic test hard is called cancellation. Cancellation occurs when two hypotheses interact relative to a particular data item. As an example, I may

have flu, which tends to increase a temperature, but I may also have a lowered immune function, which tends not to show higher temperature. As a result, I may not show high fever, but it's not because I don't have flu. It's much more because the symptoms of flu and the symptoms of lowered immune function are cancelling out each other. So, we also saw this in our initial exercise. We chose Thetadesis as the most parsimonious hypothesis for these data. But imagine if we didn't have that as an option. If we didn't have Thetadesis, we may have said that it's Betatosis, Iotalgia, and Kappacide, because the elevated A we see in Iotalgia cancels out the reduced A we see in Kappacide, which would account for our normal A levels. In general, cancel interactions are very hard to account for. In order to address these factors that make diagnosis so complex, it is useful to shift from the perspective of diagnostics solely as classification to a perspective of diagnostics as abduction.

07 - Deduction, Induction, Abduction

[Click here to watch the video](#)

Deduction: Given the rule and the cause, deduce the effect.

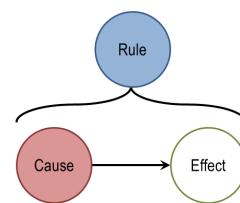


Figure 806: Deduction, Induction, Abduction

Induction: Given a cause and an effect, induce a rule.

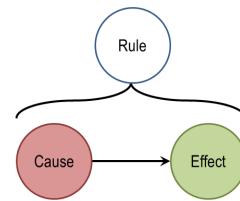


Figure 807: Deduction, Induction, Abduction

Abduction: Given a rule and an effect, abduce a cause.

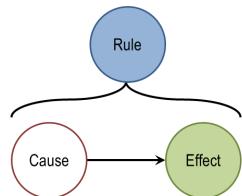


Figure 808: Deduction, Induction, Abduction

Let us look at abduction more closely. To understand the similarities and differences between deduction, abduction and induction, let us look at the relationship between a rule, cause and effect. Let's consider a simple rule like, if it is cloudy, it rains. So it is cloudy, is the cause. It rains, is the effect. If it's cloudy and it rains, it's a rule. Another example of a rule, would be the rule that Bob hates Joe. So Bob hates Joe, so whenever Joe walks in, Bob leaves. The cause is Joe walking in. The effect is Bob leaving. And the rule is Bob hates Joe. You already come across yet another instance of this particular arrangement, when we were talking about flu and fever. So the rule might be; if flu then fever. Flu is the cause. Fever is the effect. Non-deduction, in one fundamental kind of inference. We know the rule and the cause, and we need to deduce the effect. So, given the rule, if it is cloudy, then it rains, and given the cause, it is cloudy, we can deduce that it rains. Similarly, in our rule that Bob hates Joe, we can say that if Joe just walked in, we can deduce that Bob will leave. Part of the rule is, if flu, then fever. And we know that a shortcut's flu, then we can deduce that, a shortcut's fever. This is simply an instance of more despondence, and we discussed this in detail when we were talking about logic. Now let us look at induction. Given a relationship between a cause and an effect, we can try to induce a rule. For example, if we observe repeatedly that when it is cloudy, it rains, and we may induce a rule. If it is cloudy, then it rains. Same thing with Bob and Joe. If we observe repeatedly that every time Joe arrives, Bob leaves. we can induce a rule that Bob must hate Joe. If every time a patient goes to a doctor with flu, and the patient

has fever, then we can induce a rule if flu then fever. In case of abduction, given a rule and an effect, we can abduce a cause. As an example, given the rule if it is cloudy then it rains, and the effect that it is raining, we can ask ourselves is it cloudy. And once again with Bob and Joe, given our rule that Bob hates Joe, and given that we just arrived at the party, and we see that Joe is here but not Bob, we might be able to abduce that Bob left when Joe arrived. Or given the rule, if flu then fever and that fact Ashok has fever, we might be able to abduce that Ashok has flu. First of all notice that we are back to diagnosis. Diagnosis is an instance of abduction. But notice several of the properties. First, deduction is truth preserving. If the rule is true, and the cause is true, we can always guarantee that the effect is true as well. Induction and abduction are not truth-preserving. If we know something of the relations between cause and effect for some sample, that does not mean that the same relationship holds for the entire population. Induction does not always guarantee correctness. Same for abduction. We may know the rule and the effect, and we may suppose that the cause is true. But that need not necessarily be true. It may be the case the flu then fever. And Ashok may have fever, but that does not necessarily mean that a shark has flu. Fever can be caused by many many things. The reason that fever does not necessarily mean that Ashok has flu is because there can be multiple causes for the same effect. Multiple hypothesis for the same data. This is exactly the problem that we had encountered earlier, when we were talking about what makes diagnosis hard. We said that deduction, induction, and abduction, are three of the fundamental forms of inference. We can of course also combine these inferences, science is a good example, you and I as scientists, observe some data about the world. Then we induced some explanation for it. Having induced such an explanation for it, we induce a rule. Having induced a rule, now we can use production to predict new data elements. We are going to observe some more. Again, we abduce, induce, deduce. And we continue the cycle. Might the cycle also explain a significant part of cognition? Is this

what you and I do on a daily basis? Adduce, induce, deduce?

08 - Criteria for Choosing a Hypothesis

[Click here to watch the video](#)

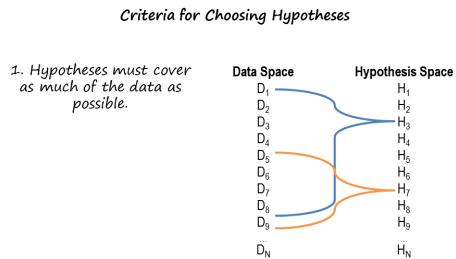


Figure 809: Criteria for Choosing a Hypothesis

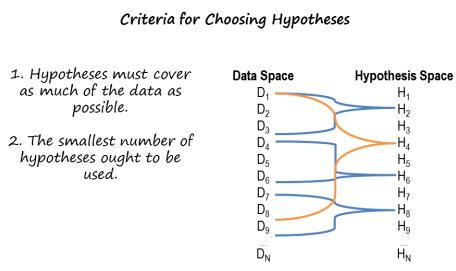


Figure 810: Criteria for Choosing a Hypothesis

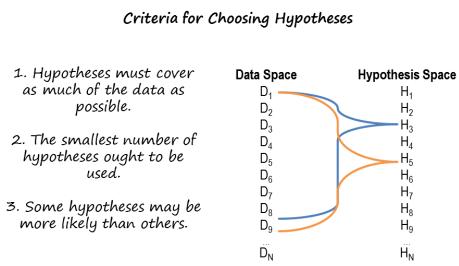


Figure 811: Criteria for Choosing a Hypothesis

Now that we understand abduction, and now that we know the diagnosis is an instance of abduction, let us ask ourselves, how does this understanding help us in choosing hypotheses? So the first principle for choosing a hypothesis is explanatory coverage. A hypotheses must cover as much of the data as possible. Here's an example, hypotheses H3 explain data items D1

through D8. Hypothesis H7 explains data item D5 to D9. Assuming that all of these data elements are equally important or equally salient, we may prefer H3 over H7 because it explains more of the data than does H7. The second principle for choosing between competing hypotheses is called the principle of Parsimony. All things being equal, we want to pick the simplest explanation for the data. So consider the following scenario. H2 explains data elements D1 to D3. H4 explains data elements D1 through D8. H6 explains data elements D4 to D6 and H8 explains data elements D7 to D9. Now if you went by the criteria of explanatory coverage, then we might pick H2, plus H6, plus H8, because the three of them combined, explain more than just H4. However, the criteria of Parsimony would suggest if you pick H4, because H4 alone, explains almost all the data, and we don't need the other three hypothesis. In general this is a balancing act between these two principles. We want to both maximize the coverage, and maximize the parsimony. Based on this particular example, we may go with H4 and H8. The two together explain all the data and in addition, the set of these two hypotheses is smaller than these set of hypotheses H2, H6, and H8. The [Ashok's] criteria for choosing between competing hypotheses is that we want to pick those hypotheses in which we have more confidence. Some hypotheses are more likely than others. You may have more confidence in some hypotheses than in others. As an example, in this particular scenario, H3 may explain data items D1 to D8 and H5 may explain more data elements from D1 to D9. So H5 also explains D9 that H3 doesn't. However, we may have more confidence in H3, and so we may pick H3 instead of H5. Once again this is a balancing act between these three criteria for choosing between competing diagnostic hypotheses. A quick point to note here, these three criteria are useful for choosing between competing hypotheses even if the task is not diagnosis. The same problem occurs for example in intelligence analysis. Imagine that you have some data that needs to be explained and your competing hypothesis for explaining that particular data, well, you may pick between the competing hypothesis based on

this criteria. All of the task is not a diagnostic task. These three criteria are useful for explanation. Diagnosis simply happens to be an example of this [Ashok's] task.

09 - Exercise Diagnosis as Abduction

[Click here to watch the video](#)

Patient:	Illnesses:
A: Normal	<input type="radio"/> Alphitis: Elevated A, Reduced C, Elevated F
B: High	<input type="radio"/> Betatosis: Elevated B, Reduced C, Elevated E, Reduced H
C: Low	<input type="radio"/> Gammaomnia: Elevated D, Elevated E, Elevated F
D: Normal	<input type="radio"/> Deltacl: Elevated B, Reduced C
E: Normal	<input type="radio"/> Episcusitis: Reduced H
F: Low	<input type="radio"/> Zetad: Elevated B, Reduced C, Reduced E, Reduced F
G: Normal	<input type="radio"/> Etaemia: Elevated A, Reduced D, Reduced H
H: Low	<input type="radio"/> Thetadesis: Elevated B, Reduced C, Reduced H
What illness (or set of illnesses) would you use to diagnose this patient?	
<input type="radio"/> Iotalgia: Elevated A, Reduced E, Elevated F, Elevated G	
<input type="radio"/> Kappacide: Reduced A, Reduced F, Reduced G	
<input type="radio"/> Lambdacrite: Reduced A, Reduced E, Reduced F, Reduced G	
<input type="radio"/> Mutension: Elevated A, Elevated G	

Figure 812: Exercise Diagnosis as Abduction

Let us do an exercise together. The data in this particular exercise, a little bit more complicated than in the previous one. On the right-hand side, I've shown a set of diseases. What disease or subset of these diseases best explains the available data?

10 - Exercise Diagnosis as Abduction

[Click here to watch the video](#)

Patient:	Illnesses:
A: Normal	<input type="radio"/> Alphitis: Elevated A, Reduced C, Elevated F
B: High	<input checked="" type="radio"/> Betatosis: Elevated B, Reduced C, Elevated E, Reduced H
C: Low	<input type="radio"/> Gammaomnia: Elevated D, Elevated E, Elevated F
D: Normal	<input type="radio"/> Deltacl: Elevated B, Reduced C
E: Normal	<input type="radio"/> Episcusitis: Reduced H
F: Low	<input type="radio"/> Zetad: Elevated B, Reduced C, Reduced E, Reduced F
G: Normal	<input type="radio"/> Etaemia: Elevated A, Reduced D, Reduced H
H: Low	<input type="radio"/> Thetadesis: Elevated B, Reduced C, Reduced H
What illness (or set of illnesses) would you use to diagnose this patient?	
<input type="radio"/> Iotalgia: Elevated A, Reduced E, Elevated F, Elevated G	
<input type="radio"/> Kappacide: Reduced A, Reduced F, Reduced G	
<input type="radio"/> Lambdacrite: Reduced A, Reduced E, Reduced F, Reduced G	
<input type="radio"/> Mutension: Elevated A, Elevated G	

Figure 813: Exercise Diagnosis as Abduction

What answer did you give, David? So, my answer is that the best explanation is a combination of Betatosis and Zetad. By combining these two illnesses, we can cover all of the data we saw over here. We saw both illnesses elevate B, which led to our High B, and both reduced C, which led to our Low C. Our patient had a normal level of E, but the effects of Betatosis

and Zetad counteract their influence on E. Then Zetad accounts for our Reduced F, and Betatosis accounts for our Reduced H. My explanation, though, heavily weights the principle of coverage. I sacrifice parsimony by having two different explanations for the sake of coverage, I now cover all the symptoms. I could have also chosen Thetadesis. Thetadesis would have explained B, C, and H. It wouldn't explain F, but it would be a simpler explanation than the combination of Betatosis and Zetad. In that case I would be sacrificing coverage for parsimony. I could also augment Thetadesis with Kappacide and Mutension. The three of them together would cover all our symptoms, but that would also be less parsimonious than Betatosis and Zetad alone. I may have chosen to do that though, if those three diseases were much more common than Betatosis and Zetad. In which case, I would have more confidence in them than just these two. It's an excellent answer, David, but how did you come up with this answer? So what I did, is I started with the data and I looked at which of these hypotheses currently matched the data best. That is to say, which hypotheses explained the most points and had the fewest conflicts. So for example, Betatosis explained the High B, the Low C, and the Low H, but it didn't explain the Low F, and it also suggested there was Elevated E. Initially, Thetadesis was the best match, it explained three symptoms, it just didn't explain the fourth. Then, based on that, I went looking for another illness that would complete the explanation. For Thetadesis, that ended up leading me to Kappacide and Mutension, but at that point, I was using three different hypotheses. So I decided to revisit one of the closer matches from my original round. I revisited Betatosis, which had two mismatches, it predicted Elevated E and didn't explain the Reduced F. And I went looking for another hypothesis that would complete that explanation. Zetad happened to have exactly those two effects. Note that one can use alternative methods for the same problem. For example, one could use case-based reasoning, and for it, we came across a problem very similar to this one previously. Suppose that the solution to that particular problem was of a level as a

case. In that particular case, B was high, C was lower, F was low, and the solution was Thetadesis. In the current problem, the additional symptom is that F is low. So case would force you to the conclusion of Thetadesis. But it will create this particular solution to also account for the additional symptom of F being low. We could do that by adding Kappacide and Mutension to Thetadesis. Case-based reasoning does, will tend to forecast the alternative set of hypothesis. One more point to note here then, note that different methods can lead to different solutions. Given different methods, how might an error agent decide which method to select? We'll return to this particular problem when we discuss meta reasoning.

11 - Completing the Process

[Click here to watch the video](#)

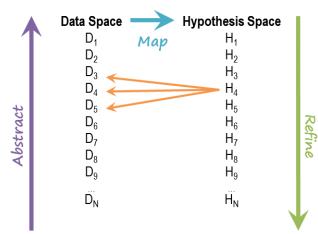


Figure 814: Completing the Process

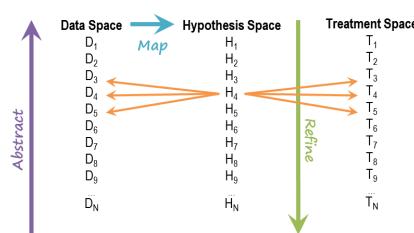


Figure 815: Completing the Process

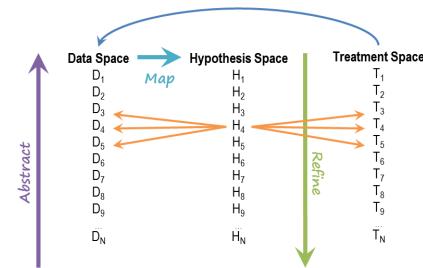


Figure 816: Completing the Process

Now we said earlier, the diagnosis is mapping from data space to hypothesis space. We also noted that although we may start with the initial set of data, proposals of specific hypotheses will lead us to collect additional data. Once we have some hypotheses for explaining the available data, then this hypotheses become indices into treatment plans. So the next step is to map the hypothesis space to the treatment space. In case of medical diagnosis, this might be a set of therapies, or a set of drugs. In case of auto mechanics the treatment space might consist of replacement of various parts or repair of various parts. Note here in the power of classification, this was the space of percepts, this was the space of actions, this mapping is very complicated. So we map the space of percepts into the equal classes in these categories. So we index the actions over the equal classes in these categories so the data index the actions of the data space. As we exit the treatment, we monitor it. And depending upon the result of this treatment, we may need to collect additional data. The treatment fails. And the fact that the treatment has failed is important data that might also lead to the collection of additional data. Thus in case of auto mechanics, if I replace a particular component and the car still does not function properly, then that is useful data to know. Because it suggests that the fault that I thought would explain the malfunction is probably not the right diagnosis. We could also think of this last phase as a type of configuration, which we talked about last time. Given a set of hypotheses about illnesses or faults with a car, we can then configure a set of treatments or repairs that best address the faults we discovered before.

12 - Assignment Diagnosis

[Click here to watch the video](#)

Assignment

How would you use diagnosis to design an agent that could answer Raven's progressive matrices?

Figure 817: Assignment Diagnosis

So would the idea of diagnosis help us design an agent that can answer Raven's progressive matrices? Perhaps the best way to think about this is to consider how your agent might respond when it answers a question wrong. First, what data will it use to investigate its incorrect answer? Second, what hypotheses might it have for incorrect answers? Third, how will it select a hypothesis that best explains that data? And last, once it's selected hypothesis that explains that data, how will it use that to repair its reasoning, so it doesn't make the same mistake again?

13 - Wrap Up

[Click here to watch the video](#)

To recap...

- Defining diagnosis
- Process of diagnosis
- Diagnosis as classification
- Diagnosis as abduction

Figure 818: Wrap Up

So today, we talked about diagnosis which is a term we're very familiar with from our everyday lives. But today, we talked about it specifically in a knowledge-based AI sense. We started off by defining diagnosis, which is finding the fault responsible for the malfunction in some

system. This can be computers, computer programs, cars or even people and animals. We then talked about the process of diagnosis, mapping data onto hypotheses and how we can see this as a form of classification. We discovered though that this can be a very complicated process and classification might not get us all the way there. So then we talked about diagnosis as a form of abduction. Given a rule and effect or a symptom, we can abduce the cause of that problem, like an illness or a software bug. Both configuration and diagnosis have been small tasks in the broader process of design. Now that we talk about them, we can talk about AI agents that can actually do design in the real world, as well as what it would mean for an AI agent to really be creative.

14 - The Cognitive Connection

[Click here to watch the video](#)

Diagnosis is a very common cognitive task. It occurs whenever our expectations are violated. We start diagnosing. Why were our expectations violated? Within a system, we expect some behavior out of it. We get a different behavior. Why did the system not give the behavior we expected from it? Notice that diagnosis is a task. We can use several methods to address it, like case-based reasoning. We have discussed diagnosis on several contexts like medicine, program debugging, car repair, but it's also very common in other aspects of our life. For example, you get unexpected traffic. Why did it occur? We review interaction with a co-worker or the economy. All are examples of diagnosis

15 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

16 - Final Quiz

[Click here to watch the video](#)

Thank you very much.

Summary

Diagnosis is like configuration in the reverse. Diagnosis is the identification of the fault(s) responsible for a malfunctioning system. Diagnosis sets-up two spaces: Data space and Hypothesis space. Data space contains data about the malfunctioning system. Hypothesis space contains data about the fault that can explain the malfunctioning system. Diagnosis is the construction of mappings from the data space to the hypothesis space. Diagnosis has two views: diagnosis as classification and diagnosis as abduction. Diagnosis is a form of abduction where, given a rule and effect or a symptom, we can abduce the cause of that problem, like an illness or a software bug. Both Configuration and Diagnosis are small tasks in the broader process of design. Diagnosis is

typically performed when our expectations of a system are violated.

References

1. Stefik, M. Introduction to Knowledge Systems,
Pages 670-680.

Optional Reading:

1. Stefik, Chapter 9; T-Square Resources
(Stefik Diagnosis _Pgs 670-690_.pdf)

Exercises

None.

Lesson 23 - Learning by Correcting Mistakes



Research is the process of going up alleys to see if they are blind.
— Marston Bates.

01 - Preview

[Click here to watch the video](#)

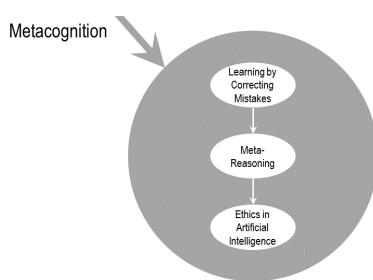


Figure 819: Preview

makes that same mistake again? As an example, I'm driving and I decide to change lanes. As I change lanes, I hear cars honking at me. Clearly I made a mistake. But what knowledge, what reasoning led to that mistake? Can I correct it, so that I don't make the mistake again? Learning to correct mistakes is our first lesson in meta-reasoning. We'll start today by revisiting explanation-based learning. Then we'll use the explanations for isolating mistakes. This will be very similar diagnosis. Except that here we'll be using explanation for isolating mistakes. This will make it clear, why explanation is so central to knowledge based AI. Then we'll talk about how we can use explanations for correcting mistakes which will set up the foundation for subsequent discussion on metareasoning.

Lesson Preview

- Explanation-based learning revisited
- Isolating mistakes
- Explaining mistakes
- Correcting mistakes

02 - Exercise Identifying a Cup

[Click here to watch the video](#)

Figure 820: Preview

Today we'll talk about another method of learning, called learning by correcting mistakes. An agent reaches a decision. The decision turns out to be incorrect, or sub optimal. Why did the agent make that mistake? Can the agent correct its own knowledge and reasoning so that it never



A Cup
A cup is an object that is stable and enables drinking.

An Object
This object is light and made of porcelain. It has a decoration, a concavity, and a handle. The bottom is flat.

Can we prove this object is a cup?

Figure 821: Exercise Identifying a Cup



Figure 822: Exercise Identifying a Cup

To illustrate learning by correcting mistakes, let's go back to an earlier example. We encountered this example when we were discussing explanation-based learning. So imagine again that you have bought a robot from the Acme hardware store, and in the morning, you told your robot, go get me a cup of coffee. Now the robot already is bootstrapped with knowledge about the definition of the cup. A cup is an object that is stable and enables drinking. The robot goes into your kitchen and can't find a single clean cup. So it looks around. This is a creative robot and it finds in the kitchen a number of other objects. One particular object has this description. The object is light and it's made of porcelain. It has decorations and it has concavity and a handle. And the bottom of this object is flat. Now the robot decides to use this object as a cup, because it can prove to itself that this object is an instance of a cup. It does so, by constructing an explanation. The explanation is based on the fact that the bottom is flat, that it has a handle, that the object is concave, and that it is light. Let us do an exercise together that will illustrate the need for learning by correcting mistakes. So shown here are six objects. And there're two questions here. The first question is, which of these objects do you think is a cup? Mark the button on the top left if you think that a particular object is a cup. The second question deals with the definition of a cup that we had in the previous screen. So mark the button on the right as solid, if you think that that particular object meets the definition of the cup in the previous screen.

03 - Exercise Identifying a Cup
[Click here to watch the video](#)



Figure 823: Exercise Identifying a Cup

Let us build on David's answers, let us suppose that the robot goes to the kitchen and finds this pail in the kitchen. It looks the pail and decides that this pail meets this definition of a cup, denoted here by the solid circle. The robot brings water to you in this pail. You look at the pail, and you say to the robot, no robot, this is not a cup. At this point you would expect a robot to learn from its failure. Cognitive agents do a lot of learning from failures. Failures are opportunities for learning. We would expect robots, and intelligent agents more generally, to learn from their failures as well. How then may a robot learn from the failure of considering this fail as a cup. Note that the problem is not limited to this particular fail. We can take a different example connecting with this particular cup. Imagine that the definition of cup included a statement that it must have a handle. In which case, the robot may not recognize that this is a cup. Later on you may teach the robot, this in fact is a good example for cup, because it's liftable. In that case, the robot will want to understand from that failure. It will want to understand why did it not consider it to be a cup? It should have considered it to be a cup. So the problem is not just about successes that turned out to be failures. But also about failures that should have been successes.

04 - Questions for Correcting Mistakes
[Click here to watch the video](#)

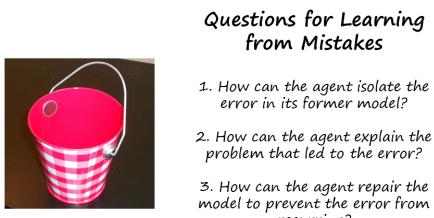


Figure 824: Questions for Correcting Mistakes

So the question then becomes, how might an error agent learn from its mistakes? Learning by correcting mistakes, or learning from failures, really entails answering three separate questions. The first question is, how can the agent isolate the error in its former model? So the agent had some model of the world. It made a mistake based on its model, how can it identify the error in its model? Note that this particular problem is very closely connected to the equation of diagnosis that we discussed earlier. The second question is how can the agent explain to itself that default it has identified the error in fact led it to the problem to the failure. Having identified the fault and explained how the fault led to the failure. The third question is how can the agent repair the fault in order to prevent the error, the failure from recurring. You may have noticed that earlier, we had related learning by correcting mistakes with matter cognition. You can see how that relationship occurs. The agent has some knowledge of the world. That knowledge leads it to failure. The agent is using that failure to repair its own knowledge. It is as if the agent is looking into itself, looking into its own reasoning, into its own knowledge and correcting itself. Note once again that the learning here is incremental. We are learning from one example at a time. However instead of simply learning from an example we are also using explanation-based learning. We're trying to explain why a particular fault led to a particular failure. An explanation connects them with explanation that's learning, not just with the notion of incremental learning. Let us see how these three questions occur in the example of the pail. The first question is how can the agent identify that the fact

that this particular pail has a moveable handle, not a fixed handle, is why this is not a good example of a cup. The second question is how can the agent with an explanation that proves why having that moveable handle makes this the poor example of a cup. Why does that lead to a failure? The third question is how can the agent change its model of a cup so that it never again picks an object with a movable handle as an example of a cup. So when we talked about explanation-based learning, we used another example of this as well. We imagined a desktop assistant that I can just say hey, fetch me that important file from last Tuesday. And it can construct its own understanding of what file I might be talking about. It has a notion of what files have been important in the past and certain criteria of those files. So constructs an understanding of what important is and tries to transfer that onto files from last Tuesday. Now imagine that I told this agent hey, fetch me that important file from last Tuesday. And it returns me a file that actually wasn't important. And I say hey, that file isn't actually important at all. The agent would first try to isolate what error it made in diagnosing that particular document as important. You might, for example, notice that every other document I ever labeled as important was very recent whereas this one was really old. So even though it met all the criteria for an important document there might be more criteria that it didn't consider yet, and one of those might be that only new documents are very important. It would then explain that the problem came from the assumption that an old document could be important, and it would then repair its model to say that in the future old documents can't be important, even if they meet the other criteria for importance. This problem identifying the error in one's knowledge that led to a failure was called credit assignment. Blame assignment might be a better term. A failure has occurred. What fault or gap in one's knowledge was responsible for the failure? That's blame assignment. In this lesson, we'll be focusing on gaps or errors in one's knowledge. In general, the error could be in one's reasoning or in one's architecture. Credit assignment applies to all of those

different kind of errors. Several Herculists, Marvin Minsky for example, consider credit assignment to be the central problem in learning. This is because error agents live in dynamic worlds. Therefore, we'll never be able to create an error agent which is perfect. Even if you were to create an error agent which had complete knowledge and perfect reasoning that lives in some world, the world around it would change over time. As it changes, the agent will start failing. Once it starts failing, it must have the ability of correcting itself, of correcting its known knowledge, correcting its own reasoning, correcting its own architecture. You can see again how this is a record of meta cognition. The agent is not diagnosing some electrical circuit or a car or software program outside. Instead, it is self diagnosing, self repairing.

05 - Visualizing Error Detection

[Click here to watch the video](#)

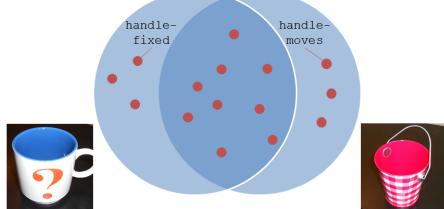


Figure 825: Visualizing Error Detection

As we mentioned previously, in journal, the editors may lie in the knowledge, in the reasoning, or the architecture of a nation. And therefore, learning and correcting errors might be applicable to any one of those. However, in this lesson, we will be focusing only on errors and knowledge. In fact, in particular, we'll be focusing on errors and classification knowledge. Classification, of course, is a topic that we have considered this class repeatedly. Let us consider an air agent that has two examples of executing an action in the world. In the first experience, the agent will use this object as a cup and gets the feedback, this indeed was a cup. So this was a positive experience. This, on the other hand, is

a negative example. Here, the agent viewed this as a cup and got the feedback that this should not have been viewed as a cup. We can visualize the problem of identifying what knowledge element led the agent to incorrectly classify this as a cup. As follows. This left circle here consists of all features that describe the positive example. The circle on the right consists of all features that describe the negative example. So features in this left circle might be things like this is a handle, there is a question mark, there is a blue interior and so on. The circle on the right consists of features that characterize a negative example. It has a movable handle. It has a red interior. It has red and white markings on the outside. There's some features that characterize only the positive experience and not the negative experience. There are those that characterize only the negative experience, and not the positive experience. There are also many features that characterize both the positive and the negative example. For example they are both concave, they both have handles, and so on. In this example, it is these features that are specially important. We'll call them fault suspicious features. We call them fault suspicious features because first they identify only the negative experience. Secondly, one or more of these features may be responsible for the fact that the agent classified this as a positive example when in fact it was a negative example. As an example suppose that this features corresponds to a movable handle. This is a false suspicious feature. It is false because this experience was false. It is suspicious because it does not characterize the positive experience. And thus it may be one of the features responsible for the fact that this was a negative example. But now there's an additional problem. There are number of false suspicious features here. So how will the agent decide which false suspicious feature to focus on? We've encountered this problem earlier, when we were talking about incremental costs of learning. At that point we had said that we wanted to give examples in an order, such that each succeeding example referred to the current constant definition, and exactly one feature. So that the agent knows exactly what the focus that feature

is on. The same kind of problem occurs again how might the agent know which feature to focus on. One possible idea is that it could try one of the features at a time and see if it will work. That is it could select this feature to repeat the process get more feedback and either is accepted or eliminated. An alternate method is that the agent perceived not just two experiences. But many such experiences. So there were other positive experiences that covered this part of the circle. That would leave only this as a false suspicious feature, and then the agent can focus attention on this feature. As an example, just like this circle may correspond to a movable handle, this may correspond to red interior. Because red interior is one of the features that characterizes a negative example and not a positive example. But later on, there might be another positive example that comes of a cup, which had a red interior in which case agent can exclude this particular feature. The reverse of this situation is also possible. Let us suppose that the agent decides that this is not a cup, perhaps because its definition says that something with a blue interior is not a cup. And therefore, it doesn't bring water to you inside this cup and tells you there is no cup available in the kitchen. You go to the kitchen. You see it and you say, well, this is the cup. Now, the agent must learn why did they decide that it was not a cup. In this case, the relevant features are these three features. These are the three features that define this cup, but do not define the other experiences. So this dot may correspond to a blue interior, this dot may correspond to a question mark on the exterior, we'll call this feature true suspicious, just like we call them false suspicious. These are the features that prevented the agent from deciding that this was a positive example of a cup. One or more of these features may be responsible for the agent's failure to recognize that this was a cup

06 - Error Detection Algorithm
[Click here to watch the video](#)

Algorithm for Isolating Mistakes

To find suspicious true-success relations:
Intersect all true successes ($\cap T$)
Union all false successes ($\cup F$)
Remove assertions in union from intersection ($\cap T - \cup F$)

To find suspicious false-success relations:
Intersect all false successes ($\cap F$)
Union all true successes ($\cup T$)
Remove all assertions in union from intersection ($\cap F - \cup T$)

Figure 826: Error Detection Algorithm

Here is an algorithm for error defined. What elements in an agent's classification knowledge may be potentially responsible for his failure? Let us look at the problem where we define the false success elements. I just said earlier. Potentially, the agent may deceive a set of positive experiences, and a set of negative experiences, not just one positive or one negative. First the intersection of all of the features that are responsible for all the false successes. A false success again is, something that we already fine a success. What was not a success like the pill, we identified it as a cup, it was not a cup. First take the intersection of all the features present for all the false successes. False-success is an object where the agent identified as a success, but in fact was false. Like the pale. The agent identified that it was a cup, but it wasn't. Then take the union of the features present for all the true successes. A true success is something that the agent classified as a success, and indeed was a success. Now remove all the assertions of the union of the true successes, from the intersection of the false successes, to identify those elements that are present in the false successes only. So, to put that differently, start off by gathering together anything that's ever present in a false success. Then, gather together everything that's true for every single true success. Remove the things that are true for every true success, from the things that are ever true for any false success. That way we get a list of only the things that are true for some false successes. So we're defining suspicious true success relationships, except that here, the operations are in reverse. So similarly, here we gather together everything that's ever true about any true success, and then gather to-

gether things that are true for every single false success. So every single false example has these things in common. Then we remove the things that every single fault example has in common, from the things that are true for any true example. As you can see, we are taking unions and intersections of features characterizing different examples. The number of examples, both positive and negative, needed for this algorithm to work well, depends on the complexity of the concept. In general, the more features you have the description of the object, more will be the number of examples we'll need, to identify the features that were responsible for our failure.

07 - Explanation-Free Repair

[Click here to watch the video](#)

Old Rule	New Rule
If: Object has bottom Bottom is flat Object has concavity Object is lightweight Object has a handle	If: Object has bottom Bottom is flat Object has concavity Object is lightweight Object has a handle Handle is fixed
Then: Object is a cup	Then: Object is a cup

Figure 827: Explanation-Free Repair

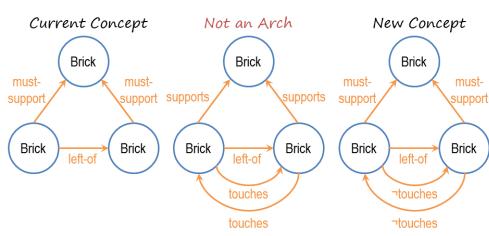


Figure 828: Explanation-Free Repair

Questions for Learning from Mistakes



1. How can the agent isolate the error in its former model?
2. How can the agent explain the problem that led to the errors?
3. How can the agent repair the model to prevent the error from recurring?

Figure 829: Explanation-Free Repair

So let us look at the result of the kind of learning technique we are discussing here. Here might be the old concept of a cup. In this particular case, this particular concept definition has been put in the form of a production rule. Here is the new, concept definition for a cup. This is almost identical to the previous definition, except that now the object not only has a handle, but also the handle is fixed. This is similar in many ways to incremental concept learning. You may recall that in incremental concept learning that any particular stage of processing there was a concept definition. As new examples came, then the concept definition changed depending on the new example and the current concept definition. Note that in this matter of concept revision, the number of features in this if clause may become very, very large, very quickly. Here we have object has a handle and handle is fixed. We could keep on adding additional features, with the interior is blue, that cover all the positive experiences. The difficulty is, at the present time there is no understanding for why the fact that handle is fixed is an important part of the cup definition. This requires an explanation. Why is it that the handle being fixed is an important part of the definition of a cup? This is one of the key differences between knowledge-based AI and other schools of AI. Classification is ubiquitous in many schools of AI as we have discussed earlier. Explanation however is a key characteristic of knowledge-based AI. Explanation leads to deeper learning. It not only says here are the features that result in a concept definition, it also says and here is why these features are important for a concept definition. This brings us to the second question on learning from fail-

ures. Now we want the agent to explain why a particular fault in its knowledge led to its failure.

08 - Explaining the Mistake

[Click here to watch the video](#)

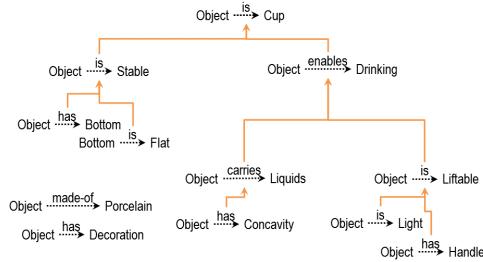


Figure 830: Explaining the Mistake

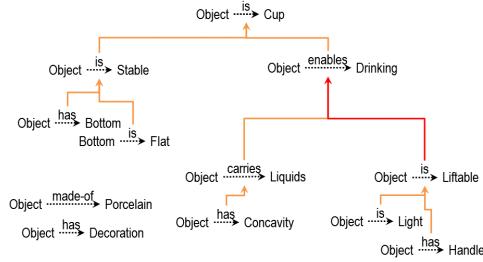


Figure 831: Explaining the Mistake

You may recall this explanation from our lesson on explanation-based learning. There, the agent constructed an explanation like this to show that a specific object was an example of a cup. For a good example of a pail, the agent may have constructed similar explanation with the object being replaced by pail everywhere. Now however, the agent knows that pail is not an example of a cup. Something is not quite right with this explanation. We've also just seen how the agent can identify the false suspicious relationship in this explanation. It is identified, but the handle must be fixed because that is the feature that separates the positive experiences from the negative experiences. The question then becomes, how can this explanation be repaired by incorporating handle as fixed? Where should handle as fixed go?

09 - Discussion Correcting the Mistake

[Click here to watch the video](#)

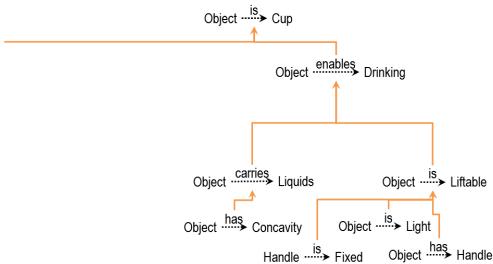


Figure 832: Discussion Correcting the Mistake

Is this a good way to fix this error?

- Yes, because it shows only fixed-handle cups enable drinking.
- No, because it will exclude some actual cups.
- No, because some non-cups will still be included.
- No, because it will cause incorrect decisions about other objects.

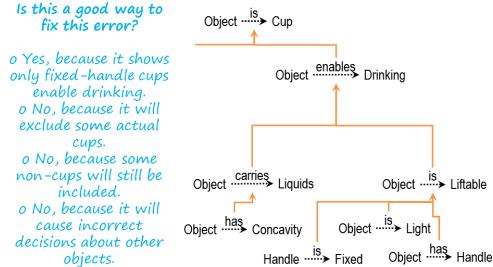


Figure 833: Discussion Correcting the Mistake

David, what do you think? Where should the agent put, Handle is Fixed, in this explanation? So it seems like the most obvious place to put it would be under the Object is Liftable connection. In order to make it liftable to enable drinking, it needs to be fixed. What do you think, is this a good way to fix the agent's error?

10 - Discussion Correcting the Mistake

[Click here to watch the video](#)

Is this a good way to fix this error?

- Yes, because it shows only fixed-handle cups enable drinking.
- No, because it will exclude some actual cups.
- No, because some non-cups will still be included.
- No, because it will cause incorrect decisions about other objects.

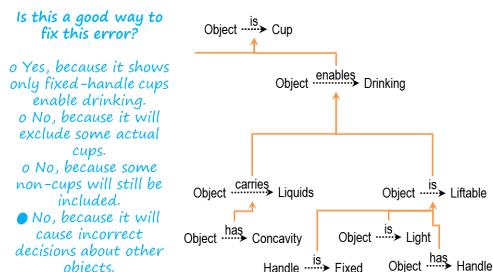


Figure 834: Discussion Correcting the Mistake

David's answer was good, but not necessarily optimal. I think he put handle as fixed

at this particular place in the explanation because the first reason here. He wanted to capture the notion, that only fixed-handle cups enable drinking. And that this spot does it. He wanted to capture the notion that only fixed-handle cups enable drinking and this scenario does that. However, this is not an optimal way of fixing this particular explanation. However, this is not the best way of fixing this particular explanation because it leads to additional decisions. It suggests that, only those things which have a fixed handle are liftable, but of course, from the pail we know, that the pail does not have a fixed handle and yet the pail is liftable. This suggests that handle is fixed should go somewhere else. Perhaps above, object is liftable, but still below object enables drinking. Agent two can figure out that this is not the optimal case for putting case as handle as fixed. Because A, it voided its notion of pail, a pail is liftable, but has a movable handle. It also voided this notion of a briefcase. This part was coming from a person of a briefcase. The briefcase with removable handle, and it to is liftable. This is how the agent knows this handle is fixed should go somewhere else, in this explanation. Above object is liftable, but beneath object is enables drinking.

11 - Correcting the Mistake

[Click here to watch the video](#)

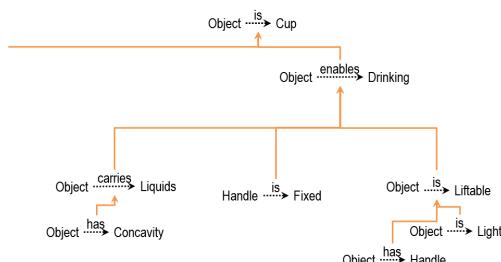


Figure 835: Correcting the Mistake

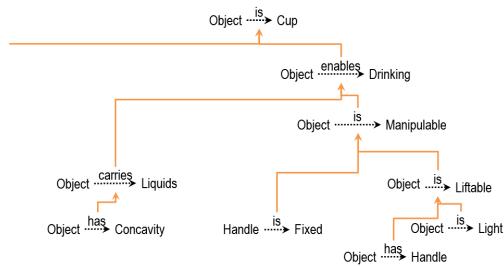


Figure 836: Correcting the Mistake

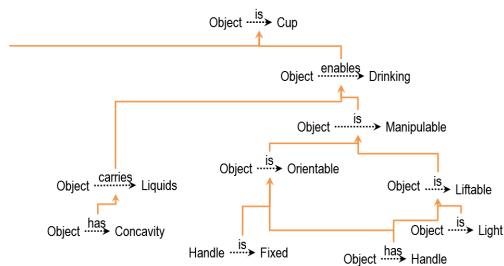


Figure 837: Correcting the Mistake

So the [recall that the] agent figured out that handle is fixed should go beneath object enables drinking, but not beneath object is liftable. So the agent will put handle is fixed here, in this particular explanation. This is correct. If the agent has background knowledge, it tells it that the reason handle is fixed is important is, because it makes the object manipulable, which in turn enables drinking. Then the agent can insert this additional assertion here in the explanation. Even more, if the agent has additional background knowledge that tells it, the defect of the object has a handle, and that the handle is fixed, together make the object orientable, which is what makes the object manipulatable, then the agent may come up with a richer explanation. The important point here is, that as powerful and important as classification is, it alone is not sufficient. There're many situations under which explanation too is very important. Explanation leads to richer learning, deeper learning.

12 - Connection to Incremental Concept Learning

[Click here to watch the video](#)

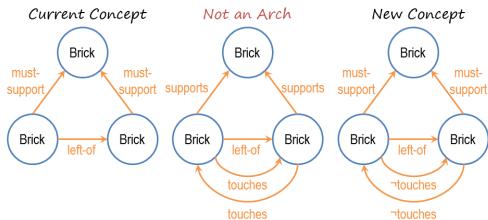


Figure 838: Connection to Incremental Concept Learning

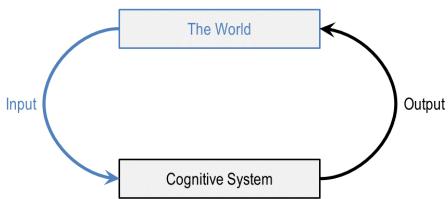


Figure 839: Connection to Incremental Concept Learning

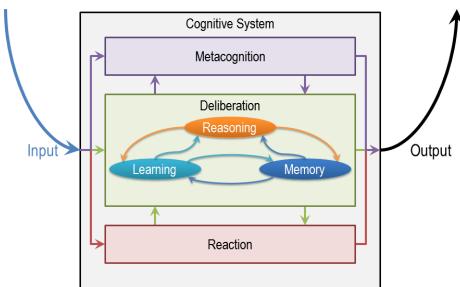


Figure 840: Connection to Incremental Concept Learning

There is one more important point to be noted here. This again is the illustration from incremental concept learning. When we were talking about incremental concept learning, we talked about technique for learning. We did not talk about how this concept were going to be used. How about in correcting mistakes, we're talking about how the agent actually uses the knowledge it learns. This point too, is centered [to Knowledge-Based AI for several] reason. The first reason is that knowledge based AI, looks at

reasoning. Looks at action, besides how knowledge is going to be used. And then, determined what knowledge is to be learned. Assess the target for learning, secondly you may recall this particular figure for the target of architecture that we'd drawn earlier. You may see that reasoning, learning and memory are closely connected, and all of that is occurring on the surface of action selection. This figure suggests that we not only learn, so that we can do action selection. But additionally, as we do action selection, and we get feedback from the world, it informs the learning. As this figure suggests, intelligent agents, cognitive systems, not only learn, so that they can take actions on the world. But further, that the world gives some feedback, and that feedback informs the learning. Once again, failure a great opportunities for learning. One additional point to be made here, learning by correcting mistakes, use learning as a problem-solving activity. An agent meets failure, it needs to learn from the failure. It converts this learning task into a problem-solving task. Let us first, identify what knowledge are related to failure. Then, let us build and explanation for this. Then we'll repair it. This learning is closely intertwined with memory, reasoning, action, and feedback from the world. Notice also, that there's reasoning, learning, and memory here. In the deliberation module closely connected with the metacognition module. Here, the reasoning, memory, and learning may be about action selection in the world. But in a metacognition module may have its own reasoning, learning, and memory capacities. And some of the learning in the metacognition is about fixing the errors in the deliberative reasoning. So, metacognition is thinking about thinking. The agent uses the knowledge, to think about the action selection, and it conducted those actions in the world. Metacognition is thinking about what went wrong in its original thinking. What was the knowledge error? We'll return to metacognition in the lesson on metareasoning.

13 - Assignment Correcting Mistakes
[Click here to watch the video](#)

Assignment
How would you use diagnosis to design an agent that could answer Raven's progressive matrices?

Figure 841: Assignment Correcting Mistakes

So how would you use learning by correcting mistakes, to design an agent that can answer Raven's progressive matrices? On one level, this might seem easy. Your agent is able to check to see if its answers are correct, so it's aware of when it makes a mistake. But the knowledge of when it's made a mistake, merely triggers the process of correcting the mistake. It doesn't correct it itself. So how will your agent isolate its mistake? What exactly is it isolating here? Once it's isolated the mistake, how will it explain the mistake? And how will that explanation then be used to correct its mistake so it doesn't make the same mistake in the future? Now in this process we can ask ourselves, will your agent correct the mistake itself?, or will you use the output to correct the mistake in your agent's reasoning? Will you look at what it did and say, here's the mistake it made. So next time it shouldn't make that mistake. If you're the one using your agents reasoning to correct your agent, then as we've asked before, who's the intelligent one? You? Or your agent?

14 - Wrap Up

[Click here to watch the video](#)

To recap...

- Explanation-based learning and incremental concept learning revisited
- Isolating mistakes
- Explaining mistakes
- Correcting mistakes

Figure 842: Wrap Up

So today, we've been talking about learning by correcting mistakes. We started off by revisiting explanation based learning and incremental concept learning, in those lessons we were dealing with a small number of examples coming in one by one. We dealt with the same thing here, but here we had the additional feedback about whether or not our initial conclusion was right or wrong. We then talk about isolating mistakes. This was similar to our problem diagnosis. Given a mistake how do we narrow down our reasoning to find where that mistake occurred. Then we talked about explain the mistake. This is key both to building human like agents and to enabling the agent to correct its mistake. Only by explaining the mistake can the agent correct it efficiently. And after those phases correction becomes a much more straightforward task. Next time we'll talk about meta-reasoning in more detail. For example here we've discussed mistakes in knowledge but what about mistakes in reasoning? What about mistakes in architecture? What about gaps instead of mistakes where we don't know something instead of knowing something incorrectly? We'll talk about all that next time.

15 - The Cognitive Connection

[Click here to watch the video](#)

Learning by correcting mistakes is a fundamental process of human learning. In fact, it may closely resemble the way you and I learn and practice. In our lives, we rarely are passive learners. Most of the time we're active participants in the learning process. Even in a difficult setting like this, you're not just listening to what I'm saying. Instead, you're using your knowledge and reasoning, to make sense of what I'm saying. You generate expectations. Sometimes those expectations may be violated. When they're violated, we generate explanations for them. We try to figure out, what was in error, in your knowledge and reasoning. This, is learning by correcting mistakes. Notice, that you think about your own thinking, a step towards meta-reasoning, which is our next lesson.

16 - Final Quiz

[Click here to watch the video](#)

Please write down what you learned in this lesson.

17 - Final Quiz

[Click here to watch the video](#)

Great. Thank you so much for your feedback.

Summary

Learning to correct mistakes is our first lesson in meta-reasoning. Learning by correcting mistakes is a fundamental process of human learning. Learning by correcting mistakes generates explanations for violations of expectations and then uses those explanations

to improve the knowledge and reasoning about the problem.

References

1. Winston P., Artificial Intelligence, Chapter 18.

Optional Reading:

1. Winston Chapter 18; [Click here](#)

Exercises

None.

Lesson 24 - Meta-Reasoning



*You see, but you do not observe.
– Sir Arthur Conan Doyle, The Adventures of Sherlock Holmes.*

01 - Preview

[Click here to watch the video](#)

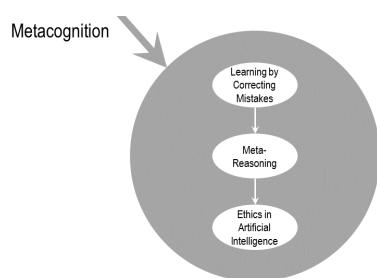


Figure 843: Preview

- Lesson Preview**
- Mistakes in knowledge, reasoning, and learning
 - Gaps in knowledge and reasoning
 - Strategy selection and integration
 - Meta-meta-reasoning?
 - Goal-based autonomy

Figure 844: Preview

Today we will talk about Meta-Reasoning. Meta-Reasoning is thinking about thinking, knowledge about knowledge. In this case, the agent does not reason about something outside in the world. Instead, the agent is reasoning about itself, but its own knowledge, its own reasoning, its own learning. As an example, I can

ask you what is President Obama's telephone number? I'm sure that all of you can immediately tell me, you don't know. But how do you know, that you don't know? We test a little bit about meta-reasoning when we were talking about learning by correcting mistakes. There, we were interested in errors in the knowledge base. In journal, errors can also be in reasoning or in learning. We'll start today by talking about mistakes in reasoning and learning, in addition to mistakes in knowledge. Then we'll talk about knowledge gaps, not just errors in the knowledge, but when some knowledge is actually missing. Then we'll talk about a very journal meta-cognitive skill, which leads to strategy selection and integration. And this class, we have talked about a large number of methods, but we haven't yet talked about how an intelligent agents can put multiple methods together to address a complex problem. Then we'll discuss meta-meta-reasoning. Or meta-meta-meta-reasoning, how far can we go? Finally, we'll discuss an example of meta-reasoning election that is sometimes called goal-based autonomy.

02 - Mistakes in Reasoning and Learning

[Click here to watch the video](#)

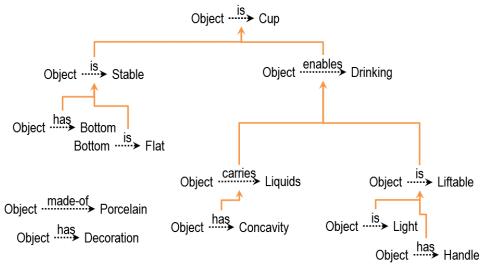


Figure 845: Mistakes in Reasoning and Learning

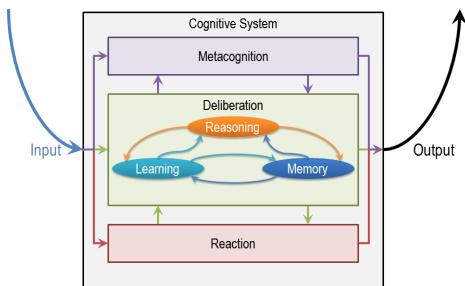


Figure 846: Mistakes in Reasoning and Learning



Figure 847: Mistakes in Reasoning and Learning

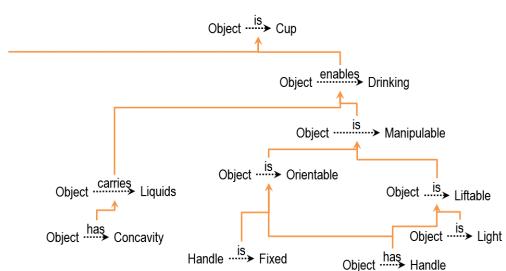


Figure 848: Mistakes in Reasoning and Learning

We have come across the notion of meta-reasoning earlier in this class. You may recall this was the explanation that an agent had built,

to decide why a particular object was an instance of the cup. And we told it that it was an incorrect decision. The agent then, reflected on its knowledge. Here was the knowledge. Here was the explanation it had built. It is now reflecting on it and trying to figure out what makes this particular explanation incorrect. Where does the fault lie? That was an example where the agent was reflecting on its knowledge, but has this knowledge was stored in the short term memory. It had been pulled out of the long term memory. But just like there can be error in the knowledge, there could also potentially be an error in reasoning. Or potentially an error in the learning process. An example of an error in reasoning occurred very early in this particular class. You may recall this particular diagram from mean sense analysis. This was the blocks micro build. The agents needed to take the blocks from this initial stage to this goal state. However there were multiple goals here. D on table, C on D, B on C. And as the agent tried to accomplish these goals. It ran into cul de sacs, where no further progress was possible without undoing some of the earlier goals. So, that was an example of metacognition over reasoning, where the agent was trying to figure out, what was the error in my reasoning, and how can I remedy it. We can similarly have metacognition over learning. So an example of metacognition where learning occurred. When we were talking about learning making mistakes. You recall this was the explanation the agent had built, after it had remedied the explanation. The remedy in this particular case was adding here that the handle is fixed and relating it to the rest of the explanation. Given that the agent had used explanation based learning to build this explanation in the first place, we can think of the agent as, reflecting on this process of explanation based learning and asking itself, what did I do wrong? And then deciding, well I built the wrong explanation. I must change that explanation. And that's what learning by correcting mistakes did. We can consider this to be a process of metacognition of learning in the sense that the agent may say, how did I learn this particular knowledge. Well, they learned it through explanationless learning. So what was

wrong in my process of explanationless learning that lead to this incorrect explanation? How do I fix my process of explanationless learning so that I do not make the same error, again?

03 - Beyond Mistakes Knowledge Gaps

[Click here to watch the video](#)

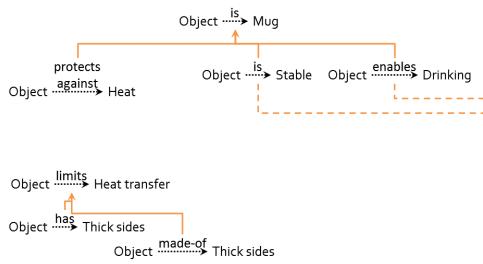


Figure 849: Beyond Mistakes Knowledge Gaps

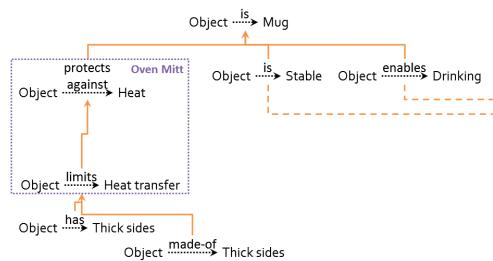


Figure 850: Beyond Mistakes Knowledge Gaps



Figure 851: Beyond Mistakes Knowledge Gaps

So far, we have talked about the case where there was an error in the knowledge or an error in the reasoning or in the learning. The knowledge for example was incorrect in some way but the knowledge can also be incomplete. There can be

a gap in knowledge or in reasoning or in learning. A gap in knowledge occur when we're doing this exercise under explanation-based learning. In that particular case, the agent had build this part of the explanation, and it also had this part of the explanation. But it could not connect these two, because there was no knowledge to connect that are object that has thick sides and it can limit heat transfer that it would protect against heat. Once the agent detects this as a knowledge gap, then it can set up a learning goal. The learning goal now, is, acquire some knowledge that will connect these two pieces of knowledge. Once the agent detects a knowledge gap, it can set up a learning goal. The learning goal now is to be able to connect these two pieces of information. Notice that we are seeing how agents can spawn goals. In this particular case the agent is spawning a learning goal. You might recall that when we did this exercise on explanation based learning, the agent went back to its memory, and found a precedent, found a piece of knowledge, that enabled it to connect these two parts of the explanation. And so this link was formed and the agent was then able to complete its explanation. This is an example how the learning goal was satisfied, using some piece of knowledge. In this case the knowledge came from the memory. But the agent could have potentially also acquired the knowledge from the external world. For example, it may have gone to a teacher and said, I have a learning goal. Help me with the knowledge that will satisfy that learning goal. Its ability to spot learning goals and then find ways of satisfying or achieving those learning goals or any goal in general, is another aspect of metacognition. So this was an example of how metacognition helps resolve a gap in knowledge. Now let us see how it can help resolve gaps in reasoning or learning. To see how metacognition can help resolve reasoning gaps, let us return to this example of using mean sense analysis in the blocks micro build. Once the agent reaches a cul de sac in the reasoning. The agent could formally list its goal and ask itself how can I help to resolve this cul-de-sac. It may then be the reminder of this strategy problem reduction was it uses its goals into sev-

eral independent goals and then the agent can go about achieving each goal at one at a time. Thus in this example, the agent set up a new reasoning goal and that used that reasoning goal to pick a different strategy and thereby achieved that reasoning goal. Note also that this is one way in which we can integrate multiple strategies. We first use some [x] analysis right in the cul-de-sac, form a new listening goal, use the listening goal to bring in a different strategy follow reduction and then go back to the original strategy means and analysis. We're achieving each goal independently

04 - The Blurred Line Between Cognition and Metacognition
[Click here to watch the video](#)

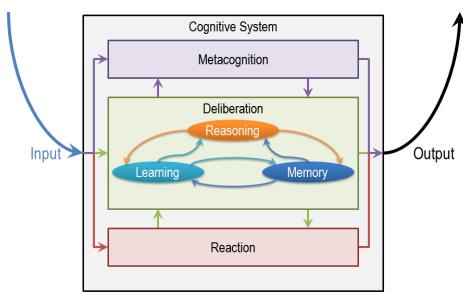


Figure 852: The Blurred Line Between Cognition and Metacognition

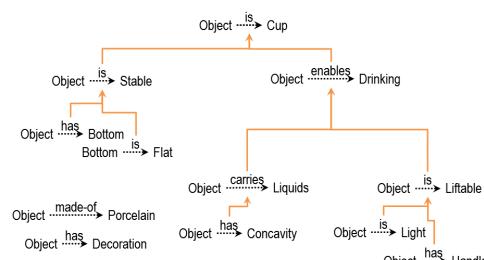


Figure 853: The Blurred Line Between Cognition and Metacognition

In this architecture for a cognitive system, we have drawn these boxes as if metacognition was completely separate from deliberation and deliberation was completely separate from reaction. In fact, there might be considerable overlap between metacognition and deliberation. Some processes in deliberation might be

viewed as metacognitive processes. Some processes are metacognitive might be viewed as deliberative processes. To see where the lines between metacognition and deliberation are blurry. Let us return to this example from explanation based learning. When we talked about explanation based learning, we did not talk about metacognition at all. We can view the agent as saying, well, I do not know how to build a connection between this part of the explanation and this part of the explanation. Therefore, I'll set up a reasoning goal which pulls at some other knowledge, and so on. Now that we know the vocabulary of megacognition, it is easy to view all of that in terms of this new vocabulary. So, instead of thinking of deliberation and metacognition as two separate independent boxes,. A better way might be, to think in terms of boxes that partially overlap, as a meta space and as a deliberation space. We should not be overly concerned, whether something should go into the deliberation space into the metacognition space. The more important thing is, what is the content of knowledge that we need to carry out a process and what is the process that we need to carry out.

05 - Strategy Selection
[Click here to watch the video](#)

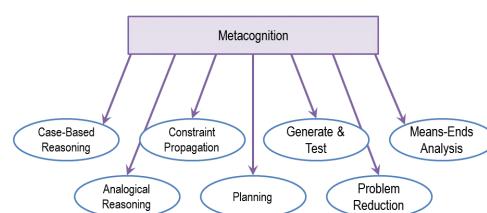


Figure 854: Strategy Selection

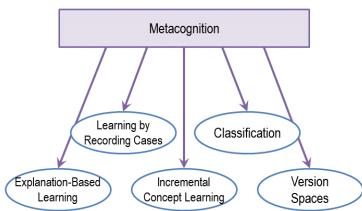


Figure 855: Strategy Selection

In this course, we have learned about a large number of reasoning methods. Here are some of them. We could have added a lot more here, for example, plan refinement or logic or scripts. Typically when you and I program an AI agent, we pick a method, and we program that method into the agent. One unanswered question is, how might an agent know about all of these methods and the autonomously select the right method for a given problem? This is the problem of strategy selection and metacognition helps with strategy selection. Given a problem, and given that all of these matters are relative to the agent to potentially address problem. Metacognition is select between these matters using several criteria. First, each of these methods require some knowledge of the world. For example, case-based reasoning requires knowledge of cases. Constraint propagation requires knowledge of constraint. And so on. Metacognition is select one particular method, depending on what knowledge is exactly available for addressing that specific input problem. If that specific input problem, case does not have a label, then clearly the method of case-based reasoning cannot be used. If, on the other hand, constraints are available, the constraint propagation might be a useful method. Second, if the knowledge required by multiple methods is available, then metacognition must select between the competing methods. Under the criteria for selecting between these methods might be computational efficiency. For a given class of problems, some of these methods might be computationally more efficient than other methods. As an example, if the problem is very close to a previously encountered case, then a case-based reasoning might be

computationally a very good method to use. On the other hand, if the new problem is very different from a previously encountered case, then case-based reasoning may not be a computationally efficient method. We've come across this issue of computational efficiency earlier in this class. For example, when we were discussing generate and test. If the problem is simple, then it is potentially possible to write a generator that will produce good solutions to it. On the other hand, for a very complex problem, the process of generating good solutions may be computationally inefficient. Similarly, if there is a single goal, then the method of means-ends analysis may be a good choice. On the other hand, if there are multiple goals that are interacting with each other, the means-ends analysis can run into all kind of cul-de-sacs, and have poor computational efficiency. A third criteria that metacognition can use to select between these various methods is quality of solutions. Some methods come with guarantees of quality of solutions. For example, logic is a method of provide some guarantees of the correctness of solutions. Thus, if this is a problem for which computational efficiency is not important, where the quality of solutions is critical, you might want to use the method of logic. Because it provides some guarantees of the quality, although it might be computationally inefficient. The same kind of analysis holds for selecting between different learning methods. Once again, given a problem, the agent may have multiple learning methods for addressing their particular problem. What method should the learning agent choose? That depends partly on the nature of the problem. Some methods are applicable to that problem, and some methods may not be applicable to that problem. Second, for example, in this learning task, if the examples come in one at a time we might use incremental concept learning. On the other hand, if all the examples are given together, then we might use decision-tree learning or identification-tree learning. Another criteria for deciding between these methods could be computational efficiency that lay down what the criteria could have to do with quality of solutions.

06 - Strategy Integration

Click here to watch the video

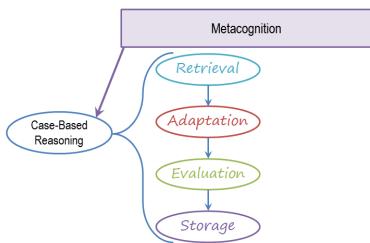


Figure 856: Strategy Integration

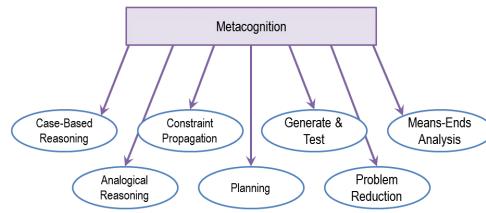


Figure 860: Strategy Integration

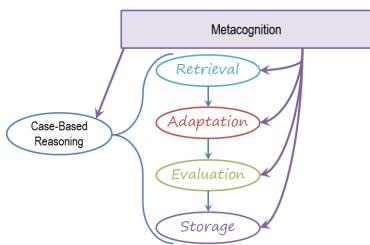


Figure 857: Strategy Integration



Figure 861: Strategy Integration

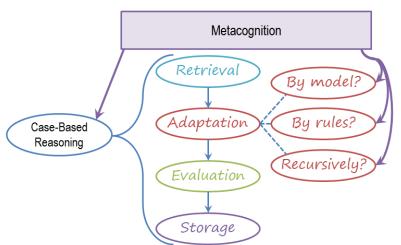


Figure 858: Strategy Integration

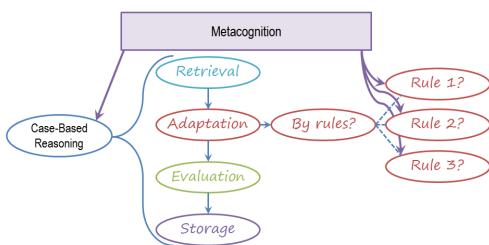


Figure 859: Strategy Integration

Now we have looked at strategy selection a little bit, let us look at strategy integration. Even if the agent selects a particular strategy, it is not necessarily stuck with that strategy. As the problem-solving evolves, it may well decide to shift from one strategy to another strategy. As an example, consider that for a given problem, metacognition decides to select the strategy of case-based reasoning. Now case-based reasoning spawns a number of sub-tasks. Retrieval, adaptation, evaluation, and storage. Metacognition can now examine the requirements for each of the sub-tasks. And then, with each of the sub-tasks, it may decide on some strategy. For example, for the task of adaptation, metacognition may pick the method of case-based reasoning recursively. Or it may pick the method of rules to adapt a case. Or it may use models for the case adaptation. If metacognition picks the method of rule based reasoning, then note that metacognition has shifted from the method of case-based reasoning overall to the method of rule-based reasoning. For a sub-task of case-based reasoning. We can also use a similar analysis at the next lower level. Suppose that metacognition decides to pick the method of rule-based reasoning for

doing the case adaptation. Now the question becomes, what rule to apply. Rule 1, 2, or 3. We can imagine meta-rules that select, which rule to apply in any given condition. We've come across a use of metacognition for strategy integration earlier. And this blocks microworld, we saw how means [means-ends analysis] can reach a cul-de-sac. When the cul-de-sac happens, metacognition may, set up a new reasoning goal and select a strategy of problem reduction for resolving the cul-de-sac. Problem reduction then, sets up four independent goals. We made it work back to mean internal assist to achieve each goal independently. In this particular case, we have integrated means and internal assistance and problem reduction and the reasoning has shifted between these two strategies in a seamless way.

07 - Process of Meta-Reasoning

[Click here to watch the video](#)

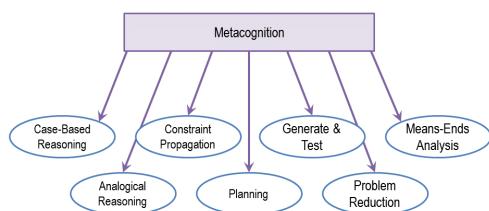


Figure 862: Process of Meta-Reasoning

That's why we have talked about some of the uses of metacognition. We can use it to fix errors in knowledge or reasoning or learning. We can use it to fix gaps in knowledge, reasoning, or learning. We can use it for strategy integration. One that we have not yet talked about is what is the processes that metacognition uses? What is its strategy? Well, metacognition can use these same kinds of reasoning processes that is the surface reasoning goal. That is, case-based reasoning is a potential strategy for metacognition. Constraint propagation is a potential strategy for metacognition, and so on. Let us take an example, supposedly if metacognition comes across a problem and it must select among these strategies. Metacognition might ask itself, what

strategy did I pick the last time I came across a similar problem? This is case-based reasoning at a meta level. To complete this example, metacognition might say the last time I used constraint propagation as a technique with this particular problem, this time I'll use constraint propagation again. As another example, given a second problem, metacognition may use planning to set up a plan for using the various strategies, thinking of each of the strategies as an operator. As a third example, metacognition might get a new problem and decide to use every single method at its disposal to generate a possible solution. It then might have some heuristics to test each one of those possible solutions and figure out which one is best. So then it's using Generate & Test to select between its multiple strategies. To summarize this part then, metacognition can use the same reasoning strategies that we have been studying at the deliberative level.

08 - Discussion Meta-Meta-Reasoning

[Click here to watch the video](#)

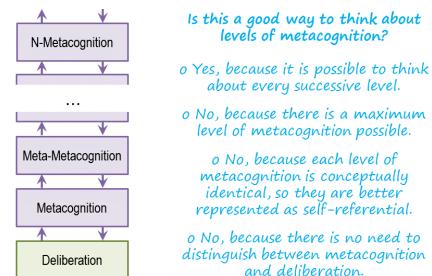


Figure 863: Discussion Meta-Meta-Reasoning

So if metacognition reasons over deliberation, could we also have an additional layer, where meta-metacognition reasons over metacognition? And to take that even further, could we have a meta-meta-metacognition reasons over meta-metacognition all the way up, infinitely up in a hierarchy? Is this a good way to think about the levels of metacognition?

09 - Discussion Meta-Meta-Reasoning

[Click here to watch the video](#)

LESSON 24 - META-REASONING

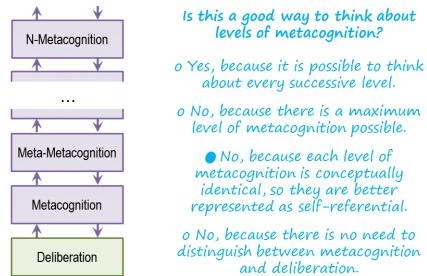


Figure 864: Discussion Meta-Meta-Reasoning

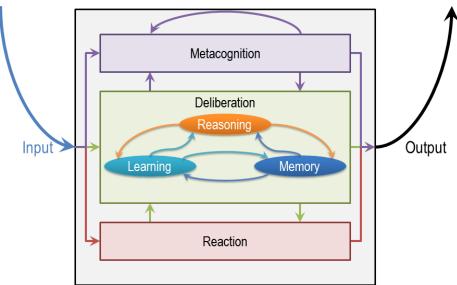


Figure 866: Example Goal-Based Autonomy

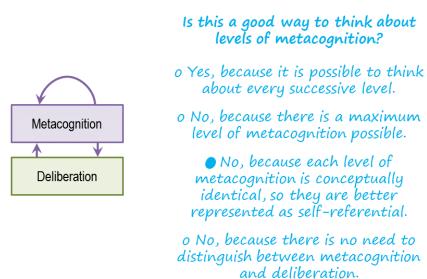


Figure 865: Discussion Meta-Meta-Reasoning

So personally, I would say no, this is not a good way to reason over it. We talked about in the previous portion of the lesson that Metacognition uses the same strategies that it reasons over. It reasons over case based reasoning. It reasons over means and analysis. It reasons over problem reduction. But it also uses case based reasoning, problem reduction, means and analysis and the others to do that reasoning. So it really doesn't need an additional layer, it's already equipped with the ability to reason over case based reasoning in the other methods, so it can just reason over itself. This is really cool, so agents don't need multiple levels of Metacognition because Metacognition reasons over itself recursively. In fact, current theories of Metacognition all talk about this kind of two layer system between Deliberation and Metacognition.

David's example of a robot that knows how to assemble cameras, but then is given the goal of disassembling a camera is a good example of goal based autonomy. Earlier we had looked at, how an agent can go about repairing his knowledge or reasoning or learning when it makes some mistake or reaches a failure. But sometimes it is not so much that the agent reaches a failure, as much as it is that the agent is given a new goal. When the agent is given a new goal, we do not want the agent to just fall apart. We do not want brittle agents. We want agents that can then adapt their reasoning methods and their learning methods to try to achieve the new goal. Even if they were not necessarily programmed to achieve that goal. We know that human cognition is very robust and flexible. You and I address a very large number of tasks, a very large number of problems and achieve a very large number of goals. If we are to design human level, human like AI agents, then those AI agents will have to be equally robust and flexible. Metacognition provides a powerful way of achieving that robustness and flexibility. It does so by flexibly, dynamically, selecting among competing strategies. It does so, reflexively and dynamically, integrating multiple strategies as the problem solving evolves. It does so, by using reasoning strategies and knowledge that were programmed into it to achieve new goals.

10 - Example Goal-Based Autonomy
[Click here to watch the video](#)

11 - Connections
[Click here to watch the video](#)

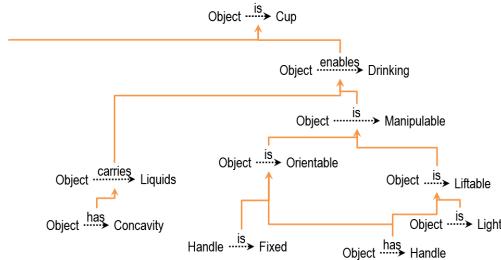


Figure 867: Connections

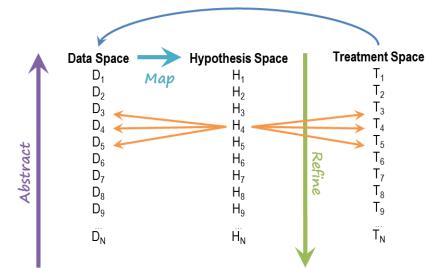


Figure 871: Connections

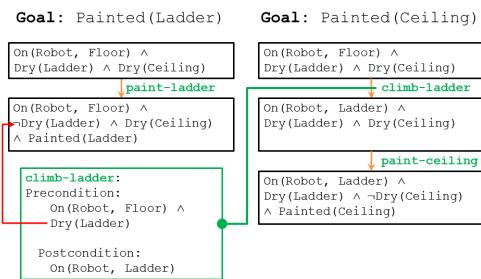


Figure 868: Connections

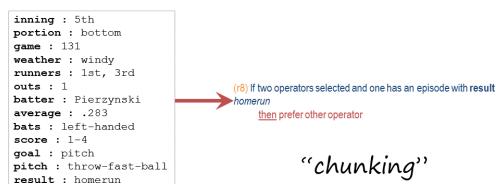


Figure 869: Connections

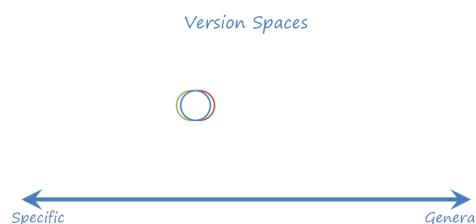


Figure 870: Connections

So, like we said earlier in this lesson, we've actually been talking about kinds of metacognition throughout this course, even if we didn't call it that at the time. We were talking about agents reflecting on their own knowledge, and correcting it when they were introduced to a mistake. Earlier in this lesson, we also talked about the possibility that an agent would reflect on the learning process that led it to the incorrect knowledge, and correct that learning process, as well. Back during partial order planning, we talked about agents that could balance multiple plans and resolve conflicts between those plans. This could be seen as a form of metacognition as well. The agent plans out a plan for achieving one goal, a plan for achieving the other goal, and then thinks about its own plans for those two goals. Then it detects the conflict between those two plans and it resolves that conflict accordingly. Then it detects the conflict between those two plans and creates a new plan to avoid that conflict. Here the agent is reasoning over its own planning process. We saw this in production systems as well. We had an agent that reached an impasse, it had two different pitches which is suggested and it couldn't decide between the two. Let's find a new learning goal to find a rule to choose between those pitches. It then selected a learning strategy, chunking, went into its memory, found a case, and chunked a rule that would resolve that impasse. In this case, the agent used that impasse to set up a new learning goal. It didn't select the strategy, strategy selection, to achieve that learning goal. We can also see medicognition in version spaces. Our agent has the notion of specific and general models, and it also has the notion of conver-

gence. The agent is consistently thinking about it's own specific and general model, and looking for opportunities to converge them down into one model of the concept. And finally, we can very clearly see metacognition in our lesson on diagnosis. We talked about how all the results for our treatment become new data for our iterative process of diagnosis. If our treatment didn't spond desirable results, it also sponds data for the metal layer. Not only do we still want to diagnose the current malfunction,. But we also want to diagnose, why we weren't able to diagnose it correctly in the first place. So, now we're diagnosing the problem with our diagnosing process. So as we can see, meta cognition's actually been implicit in several of the topics we've talked about in this course.

12 - Meta-Reasoning in CS7637

[Click here to watch the video](#)



Figure 872: Meta-Reasoning in CS7637

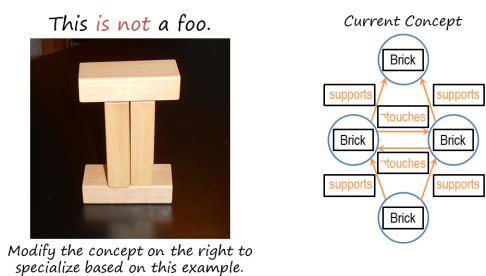


Figure 873: Meta-Reasoning in CS7637

So finally, to make things as meta as possible, meta reasoning has actually been a motivating pedagogical goal for the design of this very course. You'll notice that for almost every lesson, we start with an example of a problem that

you could solve. In incremental concept learning, for example, we start by giving you several examples of foos and not foos. And then we asked you is this a foo? In production systems, we gave you some information about a baseball game and asked you to decide what the pitcher should do next. In learning by recording cases, we gave you a world of rectangles and asked you to decide what color a new rectangle might be. In classification, we gave you a bunch of pictures and asked you to decide which of those pictures were birds. In planning, we gave you our blocks micro-world and asked you to develop a plan to go from our initial state to our goal state. In each of these, we've started with the problem that we could solve, that we then wanted to design an agent to solve. These examples then motivated our discussion not necessarily of how we did it, but how we could design an agent to do it. Then at the end of each lesson, we revisited that example. We took the reasoning method that we designed for our agent and looked at how that exact reasoning method would allow it to answer the same example with which we started the lesson. When you did the example at the start of the lesson, you didn't necessarily know how you were able to solve that problem. You could speculate, but you never know for sure. But then by building an agent that can solve that problem, we start to gain some understanding for the processes that we must be able to engage in, in order to solve that problem as well. So by designing the agent, we develop a greater understanding of our own cognition. So in this way, the very design of the lessons in this course has been driven by trying to develop metacognition in you. In fact, developing metacognition in students is the entire goal of my own PhD dissertation.

13 - Assignment Meta-Reasoning

[Click here to watch the video](#)

Assignment
How would you use meta-reasoning to design an agent that could answer Raven's progressive matrices?

Figure 874: Assignment Meta-Reasoning

So, how would you use meta-reasoning to design an agent that can answer Raven's progressive matrices? Throughout this course we've covered a wide variety of different methods for addressing this test. And each method has its own strength and its own weaknesses. Certain methods are better for some problems, and other methods for other problems. Meta-reasoning will tell us, though, that you don't have to choose just one. Your agent can have multiple methods to choose from. Discuss how you might design an agent to have meta-reasoning. What methods would it have to choose from? How will it evaluate a new problem and decide what method is best for that problem? How much improvement do you really expect to see in your agent's performance based on equipping it with meta-reasoning? And finally, will your agent engage in any kind of meta-meta-reasoning as we've discussed? Will it not only think about the methods themselves but also about how it's selecting a method? And if so, how will that improve it even further?

14 - Wrap Up[Click here to watch the video](#)**To recap...**

- Resolving mistakes and gaps
- Strategy selection and integration
- Meta-meta-reasoning?
- Goal-based autonomy

Figure 875: Wrap Up

So today we've talked about meta-reasoning. This very strongly leveraged and built on nearly everything we've talked about so far in this course. Meta-reasoning is, in many ways, reasoning about everything we've covered so far. We started off by recapping learning from correcting mistakes and the related notion of gaps. Then we covered two broad metacognitive techniques called strategy selection and strategy integration. We then discussed whether or not meta-meta-reasoning might exist. And we decided, ultimately, that such a distinction isn't even necessary. After all, the structures involved in meta-reasoning, like cases, and rules, and models, and the same as those involved in a reasoning, itself. So, meta-reasoning is already equipped to reason about itself. Finally, we discussed a particular example of meta-reasoning, called goal-based autonomy. Meta-reasoning is in many ways the capstone of our course. It covers reasoning of all the topics we've covered so far, and it provides a way that they can be used in conjunction with one another. We do have a few more things to talk about though, and we'll cover those in our Advanced Topics lesson.

15 - The Cognitive Connection[Click here to watch the video](#)

Meta reasoning arguably is one of the most critical process of the human cognition. In fact, some researchers suggest that, developing metacognitive skills at an early age in life, may be the best predictor of a student success later in life. Actually, this makes sense. Meta reasoning is not about simply learning new information, it is about learning how to learn. About, learning new reasoning strategies. About integrating new information into memory structures. Meta reasoning is also connected to creativity. In meta reasoning, the agent is monitoring its own reasoning. It is spawning goals. It is trying to achieve them. Sometimes it suspends a goal, sometimes it abandons a goal. These are all part of the creative process. Creativity is not just about creating new products. It is also about creating a processes, that lead to interesting products.

16 - Final Quiz

[Click here to watch the video](#)

So you might notice that these quizzes at the end of every single lesson have asked you to talk about what you learned in this lesson, to reflect on what you learned in this lesson. One of the goals of these final quizzes is to facilitate your own metacognition about your learning of this material as well. So by answering this, you think about your own learning, and hopefully improve it for the future. So what did you learn during this lesson?

17 - Final Quiz

[Click here to watch the video](#)

Thank you, for answering this quiz, on metacognition.

Summary

Meta-Reasoning is thinking about thinking., knowledge about knowledge. Two broad metacognitive techniques are Strategy Selection and Strategy Integration. A particular example of Meta-Reasoning is goal-based autonomy.

A Knowledge-Based Selection Mechanism for Control with Application in Design, Assembly and Planning.

2. Cox Michael, Metacognition in Computation: A selected research review.
3. Murdock William, Goel Ashok, Meta-case-based reasoning: self-improvement through self-understanding.

Optional Reading:

1. Metacognition in Computation: A selected research review; T-Square Resources (Meta Reasoning 2.pdf)
2. Meta-case-based reasoning: self-improvement through self-understanding; T-Square Resources (Meta Reasoning 3.pdf)
3. A Knowledge-Based Selection Mechanism for Control with Application in Design, Assembly, and Planning T-Square Resources (Meta Reasoning 1.pdf)
4. Metacognitive Tutoring for Inquiry-Driven Modeling; [Click here](#)

References

1. Punch William, Goel Ashok, Brown David,

Exercises

None.

Lesson 25 - Advanced Topics



Any sufficiently advanced technology is indistinguishable from magic.
– Arthur Clarke.

We are just an advanced breed of monkeys on a minor planet of a very average star. But we can understand the Universe. That makes us something very special.
– Stephen Hawking.

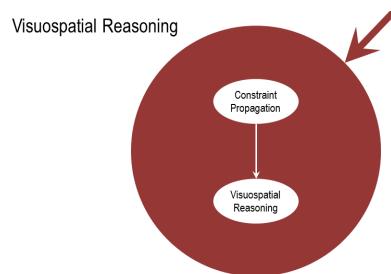
01 - Preview

[Click here to watch the video](#)



To close this class, we are talking through a handful of advanced topics, related to the course material. In this course we already discussed a variety of goals, matters and paradigms of knowledge based AI. Now let's close by talking through about some of the advanced applications of this content. We'll also talk quite a bit about some of the connections with both AI and human cognition. Many of the topics we'll discuss today are very broad and discussion oriented. So we encourage you to carry on the conversation on the forums and discuss all the issues that this content raises.

Figure 876: Visuospatial Reasoning Introduction



02 - Visuospatial Reasoning Introduction

[Click here to watch the video](#)

Figure 877: Visuospatial Reasoning Introduction



Figure 878: Visuospatial Reasoning Introduction

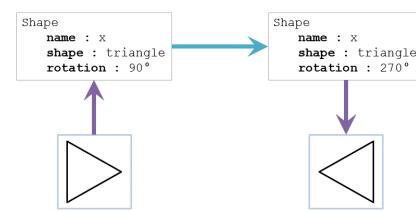


Figure 880: Two Views of Reasoning

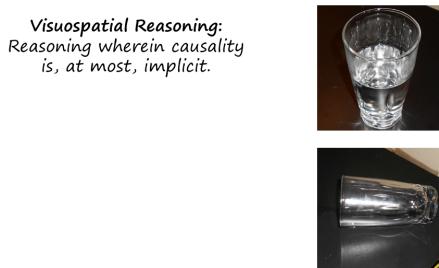


Figure 879: Visuospatial Reasoning Introduction



Figure 881: Two Views of Reasoning

Visuospatial reasoning is reasoning with visuospatial knowledge. This has two parts to it, visual and spatial. Visual deals with the what part. Spatial deals with the where part. So imagine a picture in which there is a sun on the top right of the picture. There are two parts to it. Sun, the what, the object. And where, the top right of the picture. We have come across visuospatial reasoning a little bit when we use constraint propagation to do line labeling and 2D images. One way of defining visuospatial knowledge is to say that in visuospatial knowledge causality is, at most, implicit. Imagine a picture in which there is a cup with a pool of water around it. You don't know where the pool of water came from. But you and I can quickly infer that the cup must have contained the water, and the water must have spilled out as the cup fell. So visuospatial knowledge, causality is implicit when it enables inferences about causality.

03 - Two Views of Reasoning
[Click here to watch the video](#)

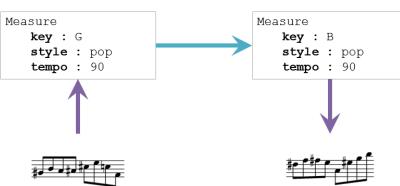


Figure 882: Two Views of Reasoning



Figure 883: Two Views of Reasoning

There're several ways of how we can deal with visuospatial knowledge. In fact in your projects you've already come across some of them. So

imagine there is a figure here. Here is a triangle with the apex facing to the right. Here is another triangle with the apex facing to the left. So in one view, the AI agent can extract propositional representations out of figures like this. And similarly propositional representations out of figures like this. So this is a propositional representation, this is a propositional representation. And then, the AI agent can work on these propositional representations to produce new propositional representations. So some AI agent can use a logic engine or a production rule to say that this particular triangle, which was rotated 90 degrees, has not been rotated to 270 degrees. So although the input wasn't in formula's figures, the action here was at the level of propositional representations of these figures. The agent may extract propositional representations like this through image processing, through image segmentation, perhaps using some techniques like constraint propagation as well. Alternatively, the agent may have analogical representations. In these analogical representations, it is a structural correspondence between the representation and the external figure. So the external world headed triangle like this, and the analogical representation will also have a triangle like this. Notice that I'm using the term Analogical Representation, we use a separate thing from analogical reasoning. We are not talking about analogical reasoning right now. We're talking about analogical representation and analogical representation is one, which is some structural correspondence with the external world that is being represented. Give a certain analogical representation, then I might want affine transformations or set transformations to get this. So I may say that I got this triangle out of that one, simply by the operation of reflection or rotation. So these proposed representations in the previous view are A model. They are separated from, divorced from the perceptual modality. These analogical representations on the other hand, are modal representations. They're very close to the perceptual modality. And human cognition, mental imagery, appears to use analogical representations. What would be an equally intuitive of computational imagery? Human cognition is

very good at using both propositional representations and analogical representations. Computers however, are not yet good at using analogical representations. Most computers, most of the time, use only prepositional presentations. The same kind of analysis may apply to other perceptual modalities, not just to our visual images. So here are two measures and we can either extract proposed representations out of them and then analyze those propositional representations. Or, we could think directly with the relationship in these two particular measures. There is a question for building queries of human cognition. When you're driving a car, and you listen to a melody on your radio and you're reminded of something. Reminded of a similar melody that you had heard earlier. What exactly is happening? Are you extracting a purpose for your presentation out of the melody that you just heard? And then the proposition representation reminds you of the proposition representation for a previously heard melody. Or, does a new melody somehow directly remind you of a previously heard melody without any intermediate propositional representation? These are our open issues in cognitive science, as well as in knowledge based AI. In cognitive science, it is by now, significant agreement that human cognition does use mental imagery at least with visual images. But we don't know how to do mental imagery in computers.

04 - Symbol Grounding Problem

[Click here to watch the video](#)

This chart summarizes some of the discussion so far. Content deals with the content of knowledge. Encoding deals with the representation of knowledge. Content and form. The content of knowledge can be visuospatial, that deals with what and where. Where is spatial, what is visual, and the encoding of the visuospatial knowledge could be either analogical or propositional. An analogical inquiry of visuospatial knowledge is a structural correspondence between the encoding and the external world that is being represented. In the propository presentation of visuospatial knowledge, there is no such correspon-

dence. Examples of this verbal knowledge include things like scripts or going to a restaurant. The script for going to a restaurant again can be represented either propositionally or potentially analogically. And a propositional presentation of the kind we say we may have tracks and props and actors. In an analogical representation of the script for going to a restaurant, we may have a short movie. In much of the codes, we have dealt with the right hand side of this chart with verbal knowledge and prepositional presentations. Part of the point of this lesson on visuospatial knowledge and reasoning is that reasoning and knowledge can be visuospatial, and representations can be analogical. But we have yet to fully understand the role of human cognition and you [we are yet to build AI agents] agents that can deal with visuospatial knowledge and analogical representation.

05 - Visuospatial Reasoning An Example

[Click here to watch the video](#)

	Visuospatial	Verbal
Content	Appearance: What and Where	Arbitrary: Driven by Inferential Needs
Encoding	Analogical: Structural Correspondence	Propositional: No Correspondence

Figure 884: Visuospatial Reasoning An Example

One area of research that does do visual spatial reasoning is called Galatia. It was developed by Jim Davies here at Georgia Tech, about 10 years back. Which is why it looks black and white and has this particular form. We provide a reference to the paper and in notes. There is a very famous problem and a logical reasoning called a Duncke problem. The Duncke problem goes something like this. First I'll tell you a story. And then I'll give you a problem. And you should try to find an answer to the problem. Let me begin with the story. Once there was a king and not a specially good king who ruled a kingdom. There was an army. That was trying to overthrow the

king. But the king lived in a fortress and it was very hard to overthrow. Moreover, the king had mined the roads, so that when the army went over the roads, it would blow off, and most of the soldiers in the army would die. The leader of the army decomposed the army into smaller groups, and these smaller groups then came to the fortress from different directions. Because each group is small enough, the mines did not blow off and each group was able to reach the fortress at the same time. They are able to overthrow the bad king. This was a story, now let me tell you about the problem. There is a patient with a cancer tumor in his body. There is a physician with a laser gun. She can use the laser gun on this tumor to kill this tumor and cure the patient. However, the laser light is so strong that it will also kill all the healthy tissue in the way, and the patient can die. What should the physician do? In most computer models of this problem, this problem is solved using propositional representations. So an example for proper surplus reduction for the original story might be that if there is a goal, and there is a resource, there is an obstacle between the resource and the goal that split the resource into many different smaller resources and bring them to the goal all at the same time but from different directions. Most composition models or decomposition problems extract some causal pattern. The causal pattern might be that if there is a goal and there is a resourcable level, and your resource can achieve the goal but there is an obstacle in the way. Then decompose the resourcing to many smaller resources and bring them to the goal in the same time from different directions. The important part here is that this is the causal pattern extracted out of the first story. Once this causal pattern has been extracted, it can be applied to this new problem. So the physician may decompose the laser beam into smaller beams and focus them on the tumor at the same time, thus curing the tumor. Jim wanted to ask whether one could do the same kind of problem solving without extracting these causal patterns. Could one use simply visual spatial knowledge? So this is visual spatial knowledge because there is both a sense of

what, the fortress, as well as where, in the middle of the figure. Notice there is visual spatial knowledge represented prepositionally. There are words here like fortress, and right road, and top road, and so on. But there is no causality that is explicit. You and I can infer the causality but it's not explicit. His Galatea program was able to find a solution to the new problem by transporting the visual spatial knowledge to the new problem, one step at a time. Thus it would map this top body part to the top rod. One here, the left body part, the left rod here, and therefore beside that, this can be decomposed. This resources, denoted by this arrow, can be decomposed into smaller resources. And then the smaller resources can arrive at this central tumor from different directions at the same time. In this way, Galatea was able to solve the addition problem without abstracting any causal pattern from it. Of course, one might say that the causal pattern is implicit here, and that is indeed true. But the entire point of a visual spatial knowledge here is that the causal pattern is not being obstructed, but as long as it is a problem-solving procedure where each step is represented only visually spatially. It is possible to transfer this problem-solving proceeded to the new crop.

06 - Visuospatial Reasoning Another Example

[Click here to watch the video](#)

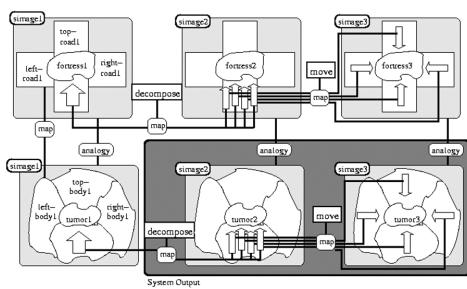


Figure 885: Visuospatial Reasoning Another Example

We just saw an example where visual spatial knowledge by itself, suffices too in our logical reasoning under certain conditions. Now let us look at a different problem. There suddenly are situations where we might want AI agents to

be able to extract [propositional] presentations. Your projects one, two, and three did exactly that. One task, where AI agent might want build proper [propositional] representations out of regional spatial knowledge is when an AI is given a design drawing. So here is a vector graphics drawing of a simple engineering system. Perhaps some of you can recognize what is happening here. This is a cylinder and this a piston. This is the rod of the piston. The piston moves. Left and right. The other end of the rod is connected to a crankshaft. As this piston moves left and right, this particular crankshaft starts moving anticlockwise. This device translates linear motion into rotational motion. I just gave you a causal account. Although because [propositional] only implicit in this [propositional] spatial knowledge. You and I were able to extract a causal account out of this. How did we do it? How can we help AI agents do it? At present if you were to make a CAD drawing using any CAD tool that you want, the machine does not understand the drawing. But can machines of tomorrow understand drawings by automatically building these causal models out of them? Put it another way. There is a story that has been captured in this particular diagram. Can a machine automatically extract the story from this diagram? In 2007, Patrick Yaner built an AI program called Archytas. Archytas was able to extract causal models out of vector graphics drawings of the kind that I just showed you. This figure is coming from paper and Archytas and hence the form of the figure. We'll have a pointer to the paper in the notes. This is how Archytas works. It began with a library of source drawings. These were drawings that we already knew about. For each drawing order it knew about it already had done the segmentation. The basic shapes for example might be things like circles and the composite shapes which were then labeled like piston and cylinder. Then a behavioral model or a causal model which said what happens when the piston moves in and out, namely the crankshaft turns. And then a functional specification we've said this particular system can work in linear motion into rotational motion. So there was a lot of knowledge with each previ-

ous drawing that Archytas already had seen. All of this knowledge was put into a library. When a new drawing was input into Archytas then it generated line segments and arcs and intersections from it. And then, it started mapping them to the lines and segments and arcs of previously known drawings. Retrieve the drawing that was the closest match in drawing to the new drawing. And then started transferring basic shapes, and then composite shapes, and it transferred each element through this abstraction hierarchy all the way up to the functional level. As an example, if Archytas library contains piston and crankshaft drawings like this along with causal functional models for them, then given a new drawing of a piston and crankshaft device Archytas will then be able to assemble a causal functional model for the new drawing. Thus Archytas extracted causal information from which spatial presentations to analogical reasoning.

07 - Ravens Progressive Matrices

[Click here to watch the video](#)

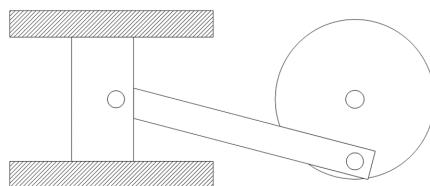


Figure 886: Ravens Progressive Matrices

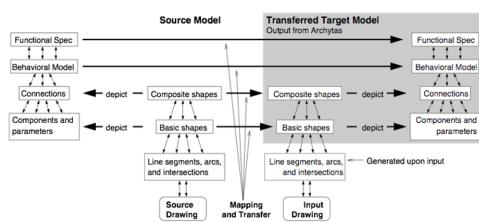


Figure 887: Ravens Progressive Matrices

Wrote another computer program that used a different kind of analogical representation called

a fractal representation. And he was able to show that the fractal representation also enables. Addressing problems from the Raven's test with a good degree of accuracy. It provides references both Maithilee's work and Keith's work in the notes.

08 - Systems Thinking Introduction

[Click here to watch the video](#)

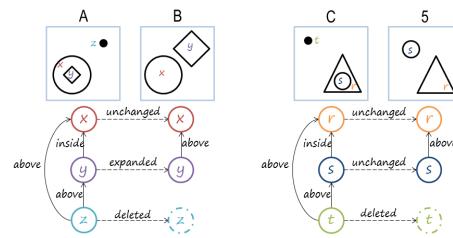


Figure 888: Systems Thinking Introduction

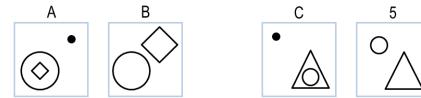


Figure 889: Systems Thinking Introduction

In this class we have talked a lot about how AI agents must be able to reason about the work. But the external work consists of systems from many different kinds. A system is composed of heterogeneous interacting components. The interaction within components, lead to processes of different kinds. These processes can occur at many different levels of abstraction. Some of the processes might be invisible. Consider an ecosystem. In an ecosystem, processes occur at many levels of abstraction. Physical, biological, chemical. Some of these processes are invisible to the naked eye, but they influence each other. Similarly in business, businesses are composed of a large number of interacting units, manufacturing, marketing, delivery, and so on. Each of

these units can be described at many levels of abstraction, from individuals, to teams, to full organizations. Given that the extra work consists of systems of different kinds. Their agents must be capable of systems thinking. They must be capable of thinking about the invisible properties of systems. About the complex behavior of the systems. In particular, they must be able to derive the invisible processes from the visible structure. This is systems thinking.

09 - Systems Thinking Connections

[Click here to watch the video](#)



Figure 890: Systems Thinking Connections

*Systems Thinking:
Reasoning about systems
with numerous components
and processes at multiple,
potentially invisible, levels of
abstraction.*

Figure 891: Systems Thinking Connections

We can connect the subject's thinking to several topics we have already covered in this class. You may recall that early on in this course, we used frames to understand stories of this kind. Now this story is actually capturing some information about a system. In this political system, there is a country with people and a president. It also had geological faults as a result of which earthquakes occur that killed some people. So the components, the complex relationships between these components, the complex processes that emerge out of this interaction between the

components. Scripts, too, are related to system thinking. Dining at a restaurant is a complex system. Once again, there are a number of components, relationships, interactions, processes, functions. A script is a knowledge for presentation that allows us to capture knowledge of a stereotypical dining experience at a restaurant. When we discussed diagnosis, at that time we talked about systems thinking a little bit more explicitly. To begin with, we said that diagnosis is identification of a fault responsible for a malfunctioning system. Diagnosis begins with the observed data about some system what are its expectations of the behavior expected of it. The diagnostic paths takes the data and matches it to hypotheses of a fault responsible for that behavior. What makes a diagnostic task so hard, again, is that there are a large number of components that interact with each other, and it is a complex behavior that emerges out of the interaction between these components. Just think of program debugging. Program debugging is hard because there are a large number of lines in the code, and these lines in the code interact with each other, and very complex behaviors, output behaviors, emerge because of those interactions. So another good example of how diagnosis is a case of systems thinking is if we look at ecological systems. Recently there's been a massive drop in the population of bees around the world. That's our data. As it turns out, the cause for this drop in bee population is the presence of a poisonous substance in insecticides used in the past several years. However, we can't see the process of bees getting poisoned by this chemical. All we can do is infer it based on higher level interactions of seeing the bee populations drop and seeing a rise in this chemical. So, in this way, there's multiple levels of abstraction in this process. There's the visible bee population, there's the visible use of pesticide, but there's also an invisible layer where the pesticide actually poisons the bees. So, we have to discern the interaction between these multiple levels. In any complex system there will be many levels of abstraction, some invisible, some visible. The human eye, or human senses more generally, can see only some of these levels of abstraction, the visible levels

of abstraction. System thinking helps us understand the invisible levels.

10 - Structure-Behavior-Function

[Click here to watch the video](#)

Today, an extremely serious earthquake of magnitude 8.5 hit Lower Slabovia, killing 25 people and causing \$500 million in damage. The President of Lower Slabovia said that the hard-hit area near the Sadie Hawkins fault has been a danger zone for years.

Today, the **President of Lower Slabovia killed 25** proposals totaling **\$500 million** for research in **earthquake prediction**. Our Lower Slabovian correspondent calculates that **8.5** research proposals are rejected for every one approved. There are rumors that the President's science advisor, **Sadie Hawkins**, is at fault.

Figure 892: Structure-Behavior-Function

Restaurant Script

```
Script
script : restaurant
track : formal dining
props : tables, menu, check,
money, F = food, P = place
roles : S = customer, W = waiter,
C = cook, M = cashier,
O = owner
entry : S is hungry, S has money
result : S has less money,
O has more money,
S is not hungry,
S is pleased
scenes :
```

Figure 893: Structure-Behavior-Function

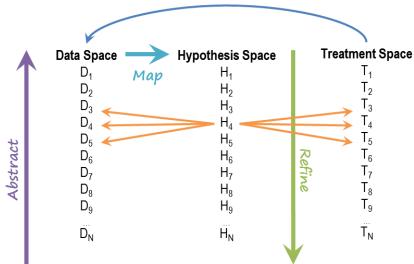


Figure 894: Structure-Behavior-Function

So AI has drawn up representations that help capture both divisible levels of obstruction structure for example. And the indivisible levels like behavior and function. Therefore these models are sometimes called structure, behavior, function. Let's take a simple example. All of us are familiar with the household flashlight. You press on the button and light comes out of the bulb.

What you can see is, the button and the bulb and the body of the flashlight. You can even open the body of the flashlight and you might see some batteries inside your flashlight. That's all you can see. But of course there is more going on here. To begin with, this particular flashlight has a function. This function is invisible. You can ascribe to it, but it's nowhere inside the body of the bulb. One level at which it is used would analyze the flashlight is, to ask ourselves what does it do. Not yet, how does it work? Just what does it do, its function. Here is a representation of the function. Here is the function, create light off that light bulb circuit, or the flashlight light bulb circuit. There is some stimulus, some external force on the switch. Initially there was no light, zero lumens and finally there is some light, 30 lumens. This captures the notion that when I press on the switch, light comes out. Here is a presentation of the structure of the flashlight. Here is a light bulb, the switch, and the battery. And they're connected. All of them are attached. Here is the invisible causal process that we're calling behaviour. We'll capture this behavior through a series of states and transitions between these states. So here is electricity initially in the battery. Then this electricity flows from the battery to the bulb, and then the bulb converts this electricity into light. But in order for this electricity to flow to the bulb, this particular switch has to be in the mode ON. The switch goes into the mode ON, when someone presses on it. Electricity is converted to light, because that's the function of the bulb. Notice that these SBF models are nested. We just gave an SBF model of the flashlight circuit. But now if you want, we can do the SBF model of the lightbulb itself. How does it create light? In this way, structure behavior function models capture not just the visible structure, but also the invisible cause of processes, the behaviors and the functions. Moreover, they capture the multiple abstraction, at the level of the flashlight, at the level of the bulb, and so on. We'll not describe it in detail here, the structure behavior function models and other similar models enables systems thinking in the context of diagnosis and design of complex systems. You're provided some read-

ings about this in the course materials, if you want to read more about it.

11 - Design Introduction

[Click here to watch the video](#)

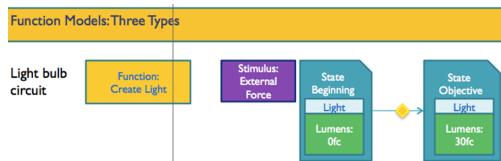


Figure 895: Design Introduction

underconstrained, open ended problems. Let's design a house that is sustainable is an example of design thinking. Sustainability here is ill-defined. The problem is open ended. In design thinking, it is not just that a solution that evolves, it is that the problem it was as well. We have problem, solution, coevolution.

12 - Agents Doing Design

[Click here to watch the video](#)

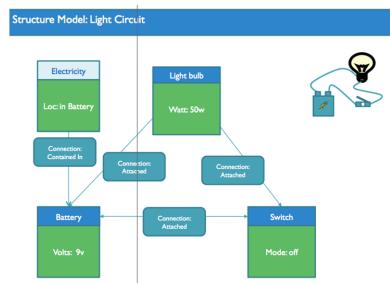


Figure 896: Design Introduction

Figure 898: Agents Doing Design

Design Thinking:
Reasoning about ill-defined,
unconstrained,
open
problems that are situated in
the world.

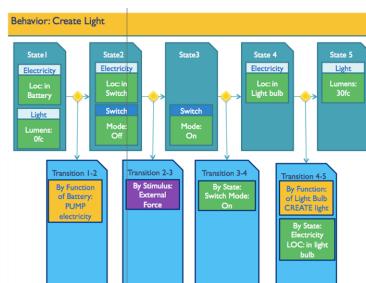


Figure 897: Design Introduction

Figure 899: Agents Doing Design

When we talked about configuration, we alluded to design. Design is a very wide ranging, open ended activity. But then we settled on to configuration, very routine kind of design, where all the parts of the design are already known, we simply have to figure out the configuration of the parts. It is time now to return to design thinking. What is design thinking? Design thinking is about thinking about ill-defined,

As we have mentioned earlier, configuration is a kind of design, a kind of routine design. And one material configuration is bound refinement. In configuration, all the components of the design are already known, but we are to find some arrangement with the components, and we assign values to some of the variables of those components, to arrive at the arrangement. Here is a design specification working it's way. Here might be a plan for designing a chair as a whole. And once we assign values to some of the variables at the level of the chair, then we can refine the plan for the chair into a plan for the chair legs, the chair seat, and so on. All of this might be subject to some constraints. There are in fact a number of AI systems, that do configuration

design. Many of them are being used in industry. Some of these AI systems use, matters like brand refinement the way we are showing it here. Others use case based reasoning. And various systems use a variety of methods, for doing configuration design, including model based reasoning and rule based reasoning. What about more creative kinds of design? Design in which not all the parts are known in advance. Since we just discussed the flashlight example, in the context of systems thinking, let us revisit that example in the context of creative design. So this is a schematic of the flashlight circuit. Here is the switch, the battery, the bulb, as earlier. On the systems thinking, we discussed how structured behavior function models capture the knowledge that when the switch is closed, electricity flows from the battery to the bulb, and the bulb converts the electrical energy into light energy. Let us suppose that this particular electrical circuit use a 1.5 volt battery and created 10 lumens of light. Tomorrow someone comes to you and says, I want 20 lumens of light. Design a flashlight electrical circuit for me. How will you do that? You might go to the structure, behavior function model for this particular circuit and do some thinking. You may recognize, the amount of light created in the bulb is directly proportional to the voltage of the battery. Instead of creating 10 lumens of light you need 20 lumens of light, you might say, I'm going to use a 3 volt battery. So far, so good. You've done system thinking in the context of design thinking. But now let us add a wrinkle. Suppose that a 3.0 volt battery is not available. At this point, a teacher tells you it's okay if a 3.0 volt battery is not available. You can connect two 1.5 volt batteries in series. Two 1.5 volt batteries connected in series will give you the voltage of three volts. Accepting the teacher's advice, you can now create an electrical circuit that will use two 1.5 volt batteries in series and create light of 20 lumens. But you're not just creating this particular design, you also learned something from it. Every design, every experience is an opportunity for learning. In the 1990s, Sam [propositional] here at Georgia Tech created a program called IDOL, IDOL did creative design. In particular, IDOL

would learn about design patterns. From simple design cases, the kind we just talked about. I'm sure most of you are familiar with the notion of design pattern, design patterns are a major construction software engineering. But design patterns are not just in software engineering but in all kinds of design, for example architecture and engineering and so on. There is some way of capturing the design pattern that can be learned from the previous case. A field of design of a device that changes the valuable variable from one value to another value. And you want another design that changes the value the same variable to some other value not the same as the previous design. One way you in which you can create the new design is. By replicating the behavior of the previous design. So not just having behavior be one for the first design, but having this behavior be one as many times as needed. Let us connect this to the example we just saw. If you have a design of an electrical circuit that can create 10 lumens of light, and you know how to do it through some behavior B1. I need to design an electrical circuit that can create 20 lumens of light, but you don't know the behavior of B2. Then this behavior B2 is a replication of behavior B1 by connecting components in series. Once Sam's program IDOL had learned about this design pattern of cascading, of replication, then, when it was given the problem of designing a water pump of higher capacity than the one available. It could create a new water pump by connecting several water pumps in series. Thus, ideal, created new designs in one domain, the domain of water pump, through analogical transfer of design patterns learned under the domain, the domain of electrical circuits. You would form the perspective of the new domain of water pumps initially did not know about all the components about all the water pumps that will be needed. With Sam's program, IDOL is creative enough to know that the pattern of problems here in the water pump is exactly the same pattern that was also occurring in the domain of electrical circuits. Sam's theory provides a computational account of not only how design patterns can be used, but also about how these design patterns can be learned and transferred to new domains. There

is of course a lot more to design. We said earlier that design thinking engages problem solution, core evolution. It's not just that a solution evolves but the problem remains fixed. But the problem evolves even as the solution evolves. It's not quite clear how humans do this kind of creative design, with this problem solution co evolution. There is certainly a few AI systems capable of problem solution coevolution at present

13 - Creativity Introduction

[Click here to watch the video](#)

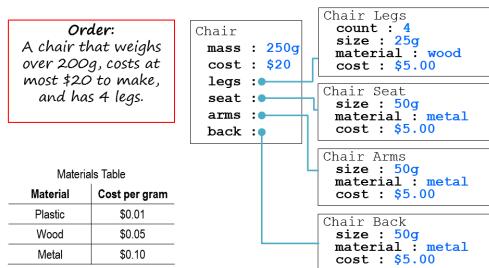


Figure 900: Creativity Introduction

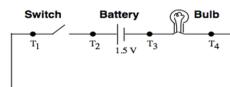


Figure 901: Creativity Introduction

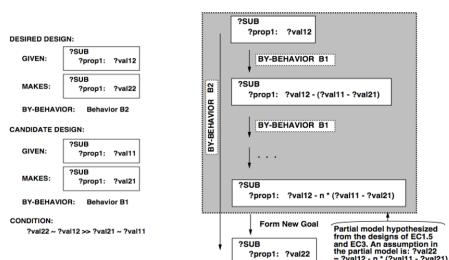


Figure 902: Creativity Introduction

This brings us to the topic of creativity. We humans are naturally very creative, and I'm not

talking just about an Einstein or a Mozart. You and I are very creative on an every day basis. You and I constantly deal with our problems. And with dealing with our problems, we don't just give up, we address them and most of the time fairly successfully. Now the goal of knowledge-based AI is to create AI agents that can think and act like humans. So shouldn't we also create knowledge-based AI agents that are creative? But in order to answer this, we have to define what is creativity. In lesson one, we saw how hard it was to define intelligence. Defining creativity is no less hard. But we will give it a try anyway.

14 - Exercise Defining Creativity I

[Click here to watch the video](#)



Figure 903: Exercise Defining Creativity I

Design & Creativity

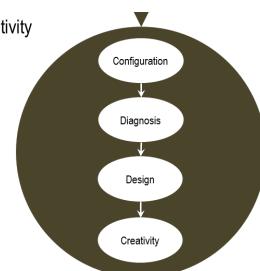


Figure 904: Exercise Defining Creativity I

In order to build AI agents that are creative, it might be useful to think about, what is creativity? Please write down your answer in this box and also post your answer in the class forum.

15 - Exercise Defining Creativity I

[Click here to watch the video](#)

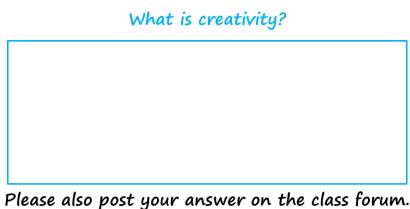


Figure 905: Exercise Defining Creativity I

So after much deliberation I decided I would define creativity simply as anything that produces an non-obvious, desirable product. I think that we have to have to sort of output for creativity in order for it to be actually be identifiable as creativity. I think that the output has to actually be wanted in some way. Doing something that no one wants is not necessarily creative. I think the output has to actually be desirable in some way, and it also has to be something non-obvious. Doing the obvious answer is not a very creative solution. If I'm propping open a door and I use a chair, it's a slightly more creative solution to that problem. Thank you David. Of course everyone's answer to this question may differ. For example, some people may not put the word product here. It's not clear that the result of creativity is necessarily a product. Some people do not put the word desirable there because sometimes creativity may not result from some initial desire. Let us carry on this discussion of what is creativity on the forum. Feel free to add your own notions.

16 - Defining Creativity II

[Click here to watch the video](#)

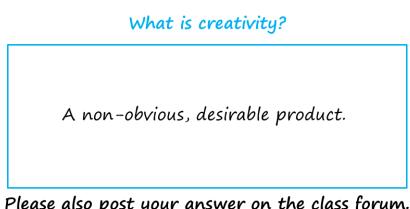


Figure 906: Defining Creativity II

Good question, David. Novelty had use with newness, the unexpectedness had use with something non-obvious or surprising. Perhaps this will become clearer if I take an example. So in my deal, we decide to entertain a group of 20 friends. We already know how to make soufflés according to a particular recipe. We'll make soufflé for 20 friends this time. We have never made soufflé for 20 people, so something is novel, something new, something we haven't done earlier. On the other hand, we have known this recipe for ages. Something unexpected would be if we come up with a new recipe for this soufflé which taste dramatically different, surprisingly different. Not just something new, but something unexpected. So far we have been talking about the product of creativity, the result of creativity, the outcome of creativity. What about the process of creativity? You use it on here some and other, both of these terms are important. Let's first look at the term other. In this course we've only talked about several processes of creativity. An analogical reasoning is a fundamental process of creativity. You already explored an analogical reasoning in the context of designing. We just did that when were talking about design thinking. One might be able to design a new kind of water pump, but composing several water pumps in series if one can analogically transfer a design factor from the domain of electrical circuits. Was a very good example. Similarly under analogical reasoning, we were talking about the processes that might be used to come up with a model the atomic structure the analogy to the model the solar system, which clearly is a creative process that cuts across large number dimensions of space and time. Another place where we talked about creative processes was when we were talking about explanation based learning. It seems creative, if the robot can go to the kitchen, and use the flower pot as a cup to bring you coffee. Here are three other processes of creativity. Emergence, re-representation, and serendipity. A simple example of emergences. If I draw three lines. One, two, three. Then a triangle emerges out of it. The triangleness doesn't belong to any single line. I was not even trying to draw a triangle.

I just drew three lines, and a triangle emerged out of it. Emergence of the triangle to the drawing the three lines is a kind of creativity. Re-representation occurs when the original representation of the problem is not conducive to problem solving. So we re-represent the problem and then commence problem solving. To see an example of this. Let's go back to atomic structure and solve this problem. Suppose that we have a model sort of system which uses the word the planets revolve around the sun. You also have a model of the atom, and this uses the term the electron rotates around the nucleus. The model of the sun had the word revolve. The model of the atom has the word rotates. The two vocabularies are different. If you were to stay with, with this couple of sort of presentations mapping between rotate and reward would be very hard. On the hand, suppose we were to re-represent this problem. Re-represent the atomic structure by growing the nucleus in the middle and the electron around it. We represent the solar system by drawing the sun in the middle and the earth around it. Then in this new representation, we can see the similarity, we can do the mock-up. This re-representation is another fundamental process of creativity. Serendipity can be of many types and can occur in many different situations. One kind of serendipity occurs when I'm trying to address a problem but I'm unable to address it. So I suspend the goal and I start doing something different. Later, at some other time, I come across a solution, and I connect it with the previous, suspended goal. The story has it that in 1941 in France, Josh Mistral's wife asked him to help her open a dress by pulling on a zipper because it was stuck. Mistral struggled with the zipper, but couldn't pull it down. Later on one day, Mistral was walking his dog, when he found that some birds were stuck to the dog's legs. Curious about this, Mistral looked at the bird closely under the microscope, he then connected the solution, the bird solution to the opening of the zipper problem and out of that was born the notion of Velcro which you and I now use on a common basis. But just like the word other was important here, these are three processes in addition to the process we already

discussed in this class. The word some is also important here. This is not an exhaustive list. There are in fact additional things we can add. For example, another potential process here is called conceptual combination.

17 - Exercise Defining Creativity III

[Click here to watch the video](#)

Something is creative if it is...

- Novel
- Valuable
- Unexpected

Some other processes of creativity:

- Emergence
- Re-representation
- Serendipity

Figure 907: Exercise Defining Creativity III

Let us do an exercise together. Here are a number of tasks that we have come across in this class. For each of these tasks, mark the box if you think that the agent that performed that task well, is a creative agent.

18 - Exercise Defining Creativity III

[Click here to watch the video](#)

For each of the following tasks, mark the box if successful completion of a task would mean that the agent is creative.

- Performing well on Raven's Progressive Matrices
- Solving the stacked blocks problem
- Configuring a chair based on input parameters
- Diagnosing an illness in a patient
- Designing a route to a new destination
- Correctly classifying new types of animals
- Using a chair to prop open a door
- Deciding between multiple strategies for a problem
- Designing a new car based on a new fuel source

Figure 908: Exercise Defining Creativity III

So actually, I marked none of them. It seems to me that for all of these tasks if an artificial agent that we design accomplishes the task, we're able to go back and look at its reasoning, look at its processing, and figure out exactly how it did it. So it's never going to be unexpected. It's always the output of a predictable algorithm. Interesting [causality was] David. Not sure I agree with it. Let's discuss it further.

19 - Exercise Defining Creativity IV

[Click here to watch the video](#)

For each of the following tasks, mark the box if successful completion of a task would mean that the agent is creative.

- Performing well on Raven's Progressive Matrices
- Solving the stacked blocks problem
- Configuring a chair based on input parameters
- Diagnosing an illness in a patient
- Designing a route to a new destination
- Correctly classifying new types of animals
- Using a chair to prop open a door
- Deciding between multiple strategies for a problem
- Designing a new car based on a new fuel source

Figure 909: Exercise Defining Creativity IV

Do you agree with David's assessment that none of these areas is creative because we can trace to the process that the agents used?

20 - Exercise Defining Creativity IV

[Click here to watch the video](#)

Do you agree with David's assessment that none of these results are creative because we can trace through the underlying process that led to them?

- Yes, because in order for a result to be creative, it must be novel, and output of an algorithm cannot be novel.
- Yes, because given a set of input, the output will always be the same; therefore, the product can never be unexpected.
- No, because it defines creativity in terms of the output rather than the process.
- No, because under this definition, humans are only considered creative because we don't know how the brain works yet.

Figure 910: Exercise Defining Creativity IV

But let's look at each of these choices, one at a time. The first one says yes, because in order for a result to be creative, it must be novel, an output of an algorithm cannot be novel. Well, there are a few problems with this particular answer. What if an output of an algorithm for a small, closed-word problem cannot be novel? The output of combinations of algorithms for open ended problem can and indeed sometimes is novel. There are algorithms for example, that do design or that do scientific discovery, whose results are novel. Let's look at the second answer. Yes, because given a set of input, the output will always be the same. Therefore, the product can never be unexpected. The output will depend not just on the input. And not only

on the methods of the system, but also the situation in which the methods are situated. The output depends not just on the input under the method the AI agent uses, but also the context in which the AI agent is situated. For example, given the same input but different context or the input, the agent will come up with very different outputs, very different understandings of that same input as we saw in this section on understanding. The third answer, no because it defines creativity in terms of the output rather than the process. This answer too has problems, because sometimes creativity can be defined simply in terms of the output without knowing anything about the process. We can think of a black box, that creates very interesting creative music. We would not know anything about the process that it is using. But, if the output's interesting and creative music, we would consider it to be creative. Personally, my sympathies lie with the fourth answer. But, of course you are welcome to disagree with me. Why don't we continue this discussion on the forum?

21 - AI Ethics

[Click here to watch the video](#)

Do you agree with David's assessment that none of these results are creative because we can trace through the underlying process that led to them?

- Yes, because in order for a result to be creative, it must be novel, and output of an algorithm cannot be novel.
- Yes, because given a set of input, the output will always be the same; therefore, the product can never be unexpected.
- No, because it defines creativity in terms of the output rather than the process.
- No, because under this definition, humans are only considered creative because we don't know how the brain works yet.

Figure 911: AI Ethics

Finally, the last topic we'll cover in this class is AI ethics. Often, scientists go about doing science without asking questions about the ethics of the science they do. We are also engrossed in questions of funding and proposals and papers. However, part of our job as scientists is to ask the question about are we doing the right kind of things? There are a large number of questions connected with the ethics of AI. We'll post a small number here today. There are no

easy answers to these questions. So I invite you to discuss these questions on the forum? First, AI Ethics put into our economy and our society. We have talked a lot in this course about designing AI agent that can act and think and behave like humans. However, in the process, we quite likely would replace some human jobs with robots. We have talked, for example, of robots that can assemble a camera. Does this mean that humans who assemble the cameras today will lose their jobs? Of course, a counter argument is that new jobs might be created, for example, jobs for designing robots. Nevertheless, there are hard issues, but ethical implications of AI in terms of human economy and society. Second, much of the modern development of AI is driven by defense applications all across the world. We already have drones, for example. It is not far-fetched to imagine a future where there are robot soldiers on the battlefield. What are the implications of introducing robot soldiers? Should we build morality into these soldiers? How do we do so? And if you are to build morality into robot, what does it teach us about our own morality? A third and related question is, that if it's hard building human characteristics like creativity and morality into AI agents, at what point do these agents become like humans? At what point do we start talking about civil rights for these machines because they're indistinguishable from humans. The idea has been touched upon in the popular culture a lot, but it is coming closer and closer to reality. What are the criteria under which we'd consider machines to be equal to humans?

Do you agree with David's assessment that none of these results are creative because we can trace through the underlying process that led to them?

- Yes, because in order for a result to be creative, it must be novel, and output of an algorithm cannot be novel.
- Yes, because given a set of input, the output will always be the same; therefore, the product can never be unexpected.
- No, because it defines creativity in terms of the output rather than the process.
- No, because under this definition, humans are only considered creative because we don't know how the brain works yet.

Figure 912: Open Issues

So today we've covered some of the most advanced and cutting edge topics in knowledge-based AI. If anything we've talked about today has really caught your eye, we invite you to check out the course materials, where we've provided several readings about each topic. As we said at the beginning, many of the things we've talked about today are open issues that the community is wrestling with right now, so we encourage you to take up the discussion over on the forums. We'll be there participating as well.

23 - Final Quiz

[Click here to watch the video](#)



Figure 913: Final Quiz

Please write down what all you learned in this lesson, in this box.

24 - Final Quiz

[Click here to watch the video](#)

Great, thank you very much.

Summary

Read about AI Ethics, an interesting topic!

22 - Open Issues

[Click here to watch the video](#)

References

Optional Reading:

1. Reasoning about Space T-Square Resources (STEFIK Visuospatial reasoning _Pgs 432-442_.pdf)
2. The Painting Fool: Stories from Building an Automated Painter T-Square Resources (Computational Creativity 1.pdf)
3. An Illustrated Conversation in Autism, Art and Creativity [Click here](#)
4. The Ethics of Artificial Intelligence T-Square Resources (AIEthics.pdf)
5. If machines are capable of doing almost any work humans can do, what will humans do? T-Square Resources (AI Ethics - 2.html)
6. OIL: Ontology Infrastructure to Enable the Semantic Web T-Square Resources (Semantic Web - 3.pdf)
7. The Semantic Web T-Square Resources (p01_theSemanticWeb.pdf)
8. Semantic Web Services T-Square Resources (Semantic Web - 2.pdf)
9. Knowledge Representation [Click here](#)
10. Prometheus Viral Clip #1 (TED Talk 2023) [Click here](#)
11. David's Birth [Click here](#)
12. Learning about Representational Modality: Design and Programming Projects for Knowledge-Based AI [Click here](#)
13. The AI Revolution: The Road to Superintelligence [Click here](#)
14. Geometry, Drawings, Visual Thinking, and Imagery: Towards a Visual Turing Test of Machine Intelligence [Click here](#)
15. Confident Reasoning on Raven's Progressive Matrices Tests [Click here](#)
16. Visual problem solving in autism, psychometrics, and AI: the case of the Raven's Progressive Matrices intelligence test [Click here](#)
17. Analogical mapping and inference with binary spatter codes and sparse distributed memory. [Click here](#)
18. A Bayesian model of rule induction in Raven's progressive matrices [Click here](#)
19. Reasoning on the Raven's advanced progressive matrices test with iconic visual representations [Click here](#)
20. Modeling multiple strategies for solving geometric analogy problems [Click here](#)
21. Solving geometric proportional analogies with the analogy model HDTP [Click here](#)
22. Raven Progressive Matrices T-Square Resources
(Raven-RavenProgressiveMatrices.pdf)

Exercises

None.

Lesson 26 - Wrap-Up



The scientific man does not aim at an immediate result. He does not expect that his advanced ideas will be readily taken up... His duty is to lay the foundation for those who are to come, and point the way.
– Nikola Tesla.

01 - Preview

[Click here to watch the video](#)

- Lesson Preview**
- Major topics revisited
 - Principles of KBAI revisited
 - Modern KBAI research
 - Final thoughts

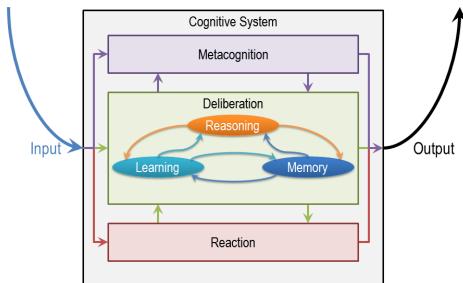


Figure 915: Cognitive Systems Revisited

Figure 914: Preview

Today we'll wrap up this course. It's been a fun journey. But like all journeys, this one too must come to an end. The goal for today's lesson is to tie together some of the big things we have been discussing, and to point to some of the other big ideas out there in the community. We'll start by revisiting the high level structure of the course. Then we'll go through some of recurrent patterns and principles we've encountered throughout the course. Finally, we'll talk about all the broader impacts and applications of knowledge based AI.

02 - Cognitive Systems Revisited

[Click here to watch the video](#)

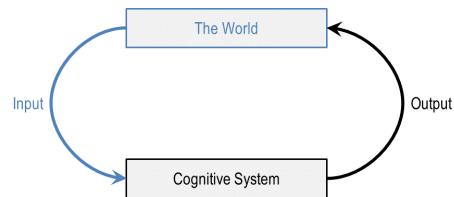


Figure 916: Cognitive Systems Revisited

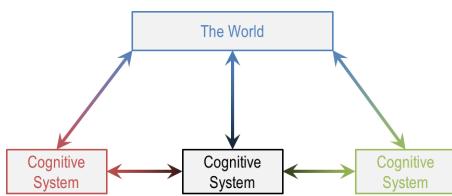


Figure 917: Cognitive Systems Revisited

Let us revisit the architecture for a cognitive system. We have come across this several times earlier in this course. We can think in terms of three different spaces. A Metacognitive space, a Deliberative space, and a Reactive space. The Reactive space directly maps percepts from the environment into actions in the environment. The Deliberative space maps the percepts into actions, with the mappings mediated by reasoning and learning and memory. So, while this is a see act cycle, this is a see think act cycle. The metacognitive space monitors the deliberative reasoning. It sees the deliberative reasoning and acts on the deliberative reasoning. As we had discussed earlier, it's better to think of these in terms of overlapping spaces rather than disjoint layers. When we were discussing metacognition we also saw that we could draw arrows back from here like this because metacognition could also act on its own. This cognitive system is situated in the world. It is getting many different kind of input from the world. Percepts, signals, goals, communication with other agents. Depending on the input, and depending upon the resources available to the agent to address that input, the agent may simply react to the input and give an output in the form of an action to a percept, for example. Or the agent may reason about the input. And then decide on an action after consulting memory and perhaps invoking learning and reasoning. In some cases, the input that the agent receives might be as a result of the output it had given to a prior input. For example, it may have received a goal. Then come up with a plan. When it had received the input of the plan, failed upon execution. In that case, deliberation can give an

alternative plan. A metacognition may wonder about, why did the planner fail in the first place? And repair the planner, not just the plan. Just like reaction and deliberation are situated in the world outside, metacognition, in a way, is situated in the world inside. It is acting on the mental world. While deliberation and reaction act on the objects in the external environment. For metacognition it is the thoughts and the learning and the reasoning that are objects internally in the amygdala. Of course, this input is coming constantly. Even now, as you and I are communicating, both you and I are receiving input. And we're giving output as well. In fact, the cognitive system is constantly situated in the world. It is never separate from the world. There is no way to separate it. It has input coming constantly. There is output going out constantly. The world is constantly changing, and the cognitive system is constantly receiving information about the world. It uses this knowledge in it's knowledge structure to make sense of the data about the world. And it decides on actions to take on the world. Recall that we had said that intelligence is a function that maps the perceptual history into action. And to some degree, intelligence is about actions, reactions. About selecting your right output to act on the world. Depending on the nature of the input the output may vary. If the input is a goal, the output might be a plan or an action. If the input is a piece of knowledge, the output might be internal to the cognitive system. And the cognitive system may assimilate that knowledge in its internal structures. If the input is a percept, the output might be an action based on that percept. If the input is new information, the cognitive system will learn from the new information, and then store the result of the learning in its memory. The world here consists not just of the physical world, but of the social world as well. Thus, this cognitive system is situated not just in the physical world but is also situated in the social world, consisting of other cognitive systems and constantly interacts with them. Once again, the interactions between these cognitive systems can be of many different kinds. Percepts, goals, actions, plans, information, data.

Any cognitive system is not just monitoring and observing the physical world, but it's also monitoring and observing the social world, the actions of other cognitive systems in the world, for example. Any cognitive system learns not just from its own actions but also from the actions of other cognitive systems around it. Thus a cognitive system needs to make sense not just to the physical world but also by the social world. We saw some of this when we were talking about scripts. There we saw how a cognitive system used a script to make sense of the actions of other cognitive systems. This architecture is not merely an architecture for building AI agents, it is also an architecture for reflecting on human cognition. Does this architecture explain large portions of human behavior?

03 - Course Structure Revisited

[Click here to watch the video](#)

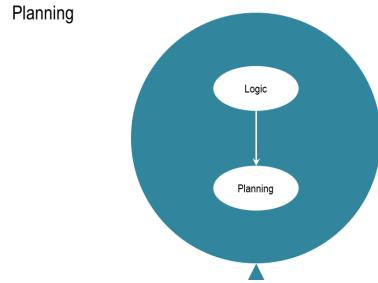


Figure 920: Course Structure Revisited

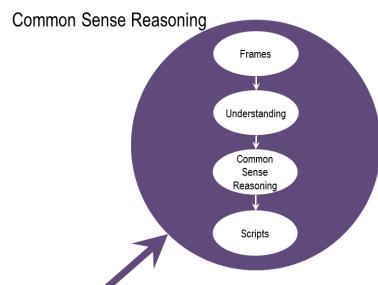


Figure 921: Course Structure Revisited

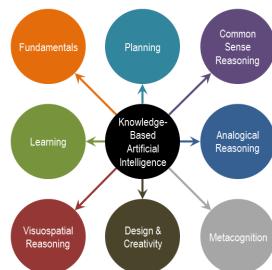


Figure 918: Course Structure Revisited

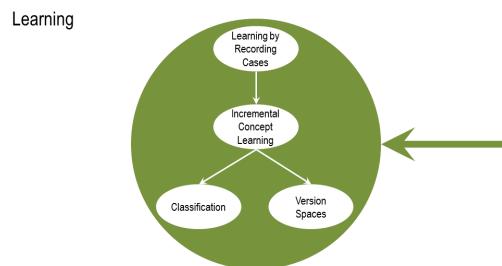


Figure 922: Course Structure Revisited

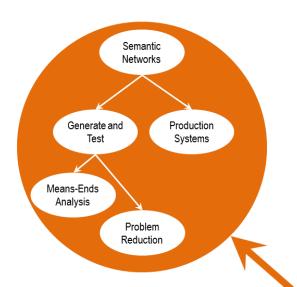


Figure 919: Course Structure Revisited

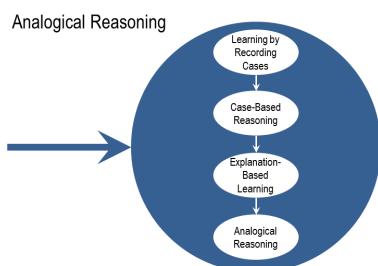


Figure 923: Course Structure Revisited

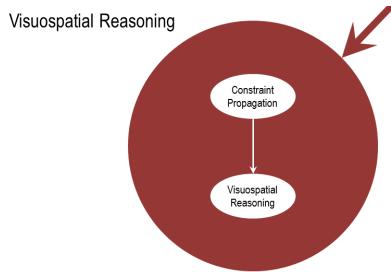


Figure 924: Course Structure Revisited

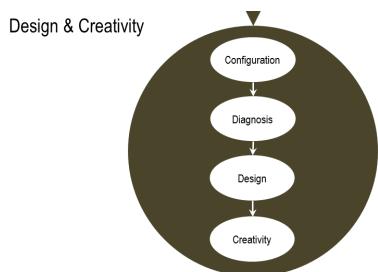


Figure 925: Course Structure Revisited

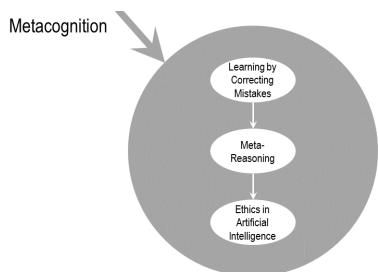


Figure 926: Course Structure Revisited

We have covered a fairly large number of topics in this course. You might recall this particular chart from the first lesson. Let us go through each one of these circles one-by-one and see the connections between them. So, in the fundamentals, we covered knowledge representations like semantic networks. In fact, we used semantic networks to address two-by-one matrix problems. Then we covered a series of problem solving methods. There's already domain independent journal purpose methods. They do not use a lot of knowledge, but they're very powerful, generate and test, means-ends analysis, and problem reduction. Then we turn to production

systems. They are a specific kind of cognitive architecture. Production systems combine reasoning, learning, and memory. Production systems to bolster technology for developing agents, and a theory of human computation. Later we talked about planning. And in order to talk about planning we first proposed a formal language for discussing the plan. It's called formal logic. We learned about the different rules and syntax used for writing formula logical statements. And learned why this is very important, that it allows agents to prove the accuracy of their conclusions based on a set of axioms. That gave us a language that we could then use to talk about planning, and as we found out actually, the origins of planning where an agent's making proofs of why a certain set of actions would lead to a certain goal. Here we talked about both partial order planning and hierarchical planning, which are two planning methods that allow agents to make advanced plans for complex tasks. And also allows us to reflect on the way we ourselves plan our actions to complex environments. Under common sense reasoning we covered several important lessons. Frames, understanding, common sense reasoning and scripts. Frames is a structured knowledge to that allows us to do understanding. Understanding is such a common every day activity. We are trying to make sense of the world, but the world is ambiguous. The word, take, for example can have so many different meanings. We saw how frames allow us to disambiguate different meanings of the word take. Under common sense reasoning we made every day intuitive inferences about the world around us. With understanding and common sense reasoning, we're concerned with sentence-level understanding. Scripts is more concerned about discourse-level understanding. Scripts is an even larger, just structure knowledge presentation than frames. It, too allows us to make sense of the world around us, both the physical world and the social world. Frames also came up in some of the other lessons, and particularly came up in the lesson on production systems, when we were trying to represent episodic knowledge. It also came up in the lesson on configuration, when we were trying to represent

plans, and the variables of various plans that can take values. We talked about learning in many lessons throughout this course. But certain lessons were explicitly and solely concerned with learning. We started off by talking about learning by recording cases, where an agent could build up a case library of its own prior experiences to use for future reasoning. That formed a foundation of analogical reasoning as well, that we'll talk about in a second. We also talked about incremental concept learning and version spaces, two different learning methods for learning about information that's coming in incrementally, or bit by bit. That was one of our foundational principles of this class that we discussed at the beginning and that we'll revisit in a few minutes. We also talked about classification under learning, which is one of the most ubiquitous problems in AI. We talked about how classification involves grouping large combinations of percepts into equivalence classes, to allow for easier action selection. Learning also can open several other lessons that we did throughout this class, including production systems, key spaced reasoning, explanation based learning, analogical reasoning, and learning about correcting mistakes. Metareasoning was also deeply connected to learning, where we could learn from our own experiences by analyzing our own prior thought processes. As David just said, analogical reasoning is another major topic connected with learning. We talked about learning by recording cases. We are simply assimilating cases by recording them in a growing case library. As a new problem comes, we use methods like nearest neighbor to retrieve the closest case, and it invoked that case with a new problem. We talked about case-based reasoning, in which we not only retrieved the case, but we also tweak it, or modify it in small ways in order to achieve a new goal. In explanation-based learning, we saw how we could connect instances to kinds of definitions. We constructed explanations that told us how instance is an example of a given concept. This involved both abstraction and transfer. We also connected explanation-based learning to creativity. Analogical reasoning made [the role of] abstraction and transfer even more explicit. We

talked about cross analogical transfer for example from the solar system to the atomic structure. We saw that when needed, rich mental models in order to be able to do an analogical transfer. We concluded the lesson on analogical reasoning by connecting it with analogical design or design by analogy. And analogical reasoning too is closely connected to creativity. Our discussion of visuospatial reasoning started with our unit on constraint propagation. Constraint propagation was a very abstract and general way of propagating constraints to make sense of a new situation, and it comes up uniquely in visual reasoning. Or we use line labeling to make sense of 3D scenes even if they're presented in 2D. Where we use line labeling to make sense of three dimensional scenes even if they're presented in only two dimensions. We also how constraint propagation can be used for natural language understanding. And we referenced how we might also use it to understand music or tactile information. Then in visuospatial reasoning we expanded on these notions to discuss whether it would be possible to reason about the world without extracting propositions. This involved looking at scenes in the world where there was no explicit causality, like a glass that's been spilled on the table. We can infer what happened, but there's nothing in the visual scene that tells us exactly how it arose. Similarly, we also discuss how this could apply to other modalities, like tactile information or musical information. In the unit on design and creativity, we considered lessons in configuration, diagnosis, design and creativity. You may recall that in configuration, we were worried about very routine, everyday kind of designs, and we did that kind of design by having a library of clients at different levels of abstraction. We selected a plan at a high level of abstraction, assign values to some of the variables, and then refine the plan at the next door level abstraction. All the components were known, we had to decide on the arrangement of the components of configuration. In diagnosis, we were given data about a malfunctioning system and we had to identify the fault responsible for that malfunctioning. We took two views of diagnosis, classification and abduction. In the classi-

fication view, this data was mapped into equal classes that acted like hypotheses for this data. In abduction, we composed this elementary hypothesis into a composite hypothesis that could best explain the entire data. Design thinking refers to thinking about problems that are ill-defined and open-ended, and under-constrained. In design thinking, the problem and solution often co-evolve. The problem doesn't remain fixed. The solution evolves, but that in turns it leads to improving the understanding of the problem, both our understanding of the problem and solution evolve together. We cut across creativity in terms of novelty, value, and the non-obviousness of the results. We discuss the criteria under which we would consider an agent to be creative. And we saw how many of the techniques that you have learned in this particular class can compose the fundamental processes of creativity. Like analogical reasoning, like visual special reasoning, like meta-reasoning. Then we close the class by talking about meta-cognition. Metacognition enable the agents to reason about their own reasoning, or think about their own thinking, or have knowledge of their own knowledge base. We started out by talking about learning by correcting mistakes. Or an agent can look at a mistake that's been made in the past, isolate the mistake, explain the mistake and then fix the mistake so that it didn't happen again. This was one narrow instance of the broader idea of meta-reasoning. In meta-reasoning, we talked about metacognition could bring together many of the different methods that we talked about throughout this class. Meta-reasoning enabled an agent to look at a new problem and select which of its many strategies would be best for addressing that problem. It also would allow an agent to integrate multiple different methods of reasoning at different levels of distraction. We also talked about what how meta-reasoning operates is also the way in which meta-reasoning operates. Meta-reasoning can reason about case based reasoning, or it can reason using case based reasoning. It could, for example, use production rules to conduct case based reasoning. In this way it could integrate many different methods at many different levels of instruction. Finally, this

set up a notion of ethics in artificial intelligence. As we're building AI agents that are starting to have real, human level intelligence, what are the ethical issues? What should we think about replacing certain human jobs with robots, or what should we think about developing robots that can interact with us on an everyday basis in the natural world? Under what conditions would we consider these agents to actually be human-like? This summarizes 30 topics that we covered in this class, which is quite a lot. Of course, there is a lot more to talk about each of these 30 topics than we have covered so far. Therefore we have provided readings for each of the topics. And you are welcome to pursue the readings for whatever topic that interests you the most. We would also love to hear about your views about this on the forum.

04 - The First Principle

[Click here to watch the video](#)

Principle #1: KBAI agents represent and organize knowledge into knowledge structures to guide and support reasoning.

Figure 927: The First Principle

Principle #1: KBAI agents represent and organize knowledge into knowledge structures to guide and support reasoning.

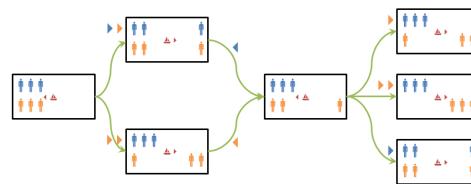


Figure 928: The First Principle

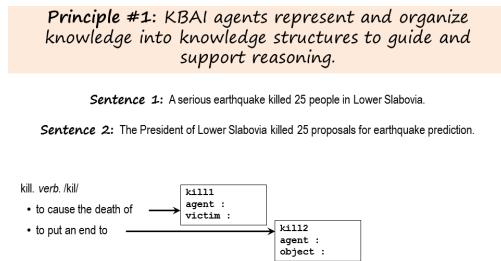


Figure 929: The First Principle

At the beginning of this course we enumerated seven major principles of knowledge based AI agents that we'll cover in CS 7637. Now let's wrap up this course by revisiting each of the seven principles. Here is the first one. Knowledge based AI agents represent and organize knowledge into knowledge structures to guide and support reasoning. So the basic paradigm here is represent and reason. Represent and reason. If you want to reason about the world, you'd have to represent knowledge about the world in some way. You not only want to represent knowledge to support reasoning, you also want to organize this knowledge into knowledge structures to guide, to focus the reasoning. Let us look at a few examples that we covered in this course about dispensing. Semantic networks not only allow us to represent knowledge about the world, they also allows us to organize that knowledge in the form of a network. We use semantic networks to address the problem of the guards and prisoners dilemma. The advantage of the semantic network was that they expose the constraints of this problems so clearly, so that we can in fact reason about it. And notice that the organization helps us focus the reasoning. Because of the organization, there's so many other choices we don't have even have to reason about them. Frames were on to the knowledge structure that organize knowledge, and guided and supported reasoning. Given frames for things like earthquakes, we could reason about sentences like, a serious earthquake killed 25 people in a particular country. We'll also use frames to support common sense reasoning. Here, Ashok is moving his body part to a sitting position. Here, Ashok is moving himself into a sitting position. Here,

Andrew sees Ashok. Now Andrew moves to the same place as Ashok, and Andrew then moves in menu to Ashok. This is about a story about visiting a restaurant. Once again, there are knowledge structures here. These knowledge structures are not only representing knowledge, they are organizing knowledge into a sequence of actions. These knowledge structures help generate expectations. So we know what Ashok expects to happen next in any of these situations. We also know how Ashok can detect surprises. When the non-obvious thing happens, Ashok knows that it has warranted the expectations of the scripts, and can do something about it. This is how the script support in guided reasoning. We also saw this principle in action, when we were talking what explanation based learning. In order to show that an instance was an example of a particular concept, cup, we constructed complex explanations. In this case, we were constructing the complex knowledge structure on the fly out of smaller knowledge structures. The smallest knowledge structures came out of precedents, or examples we had already known. Then we composed the knowledge of these various knowledge structures, into a complex explanation to doable reasoning, to guide and support their reasoning. You've seen this principle in action in several other places in this course. This is one of the fundamental principles. Represent, organize, reason.

05 - The Second Principle

[Click here to watch the video](#)

Principle #2: Learning in KBAI agents is often incremental.

Figure 930: The Second Principle

LESSON 26 - WRAP-UP

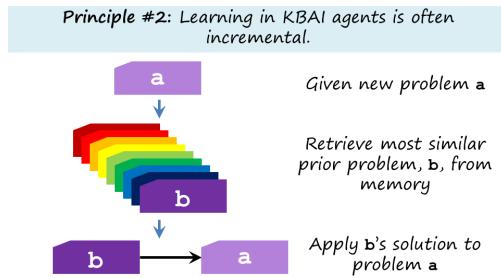


Figure 931: The Second Principle

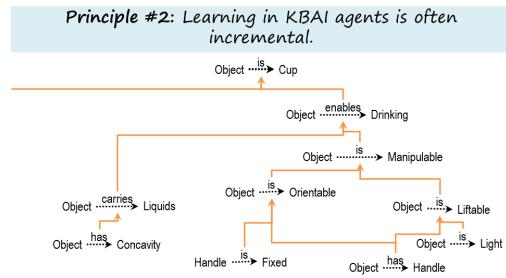


Figure 935: The Second Principle

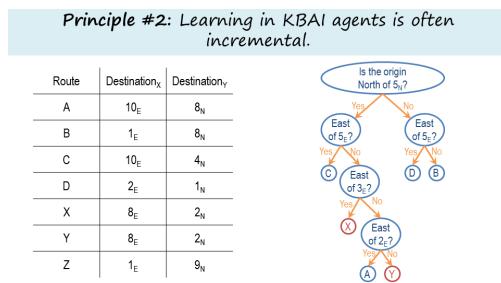


Figure 932: The Second Principle

06 - The Third Principle

[Click here to watch the video](#)

Principle #3: Reasoning in KBAI agents is top-down as well as bottom-up.

Figure 936: The Third Principle

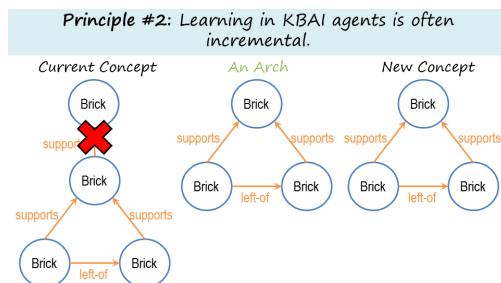


Figure 933: The Second Principle

Principle #3: Reasoning in KBAI agents is top-down as well as bottom-up.

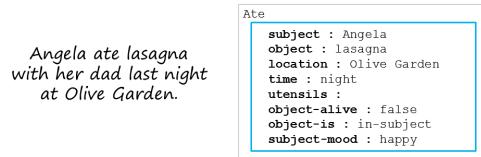


Figure 937: The Third Principle

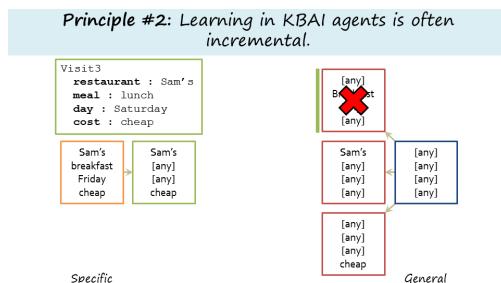


Figure 934: The Second Principle

Principle #3: Reasoning in KBAI agents is top-down as well as bottom-up.

Sentence 1: A serious earthquake killed 25 people in Lower Slabovia.
Sentence 2: The President of Lower Slabovia killed 25 proposals for earthquake prediction.

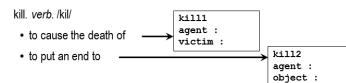


Figure 938: The Third Principle

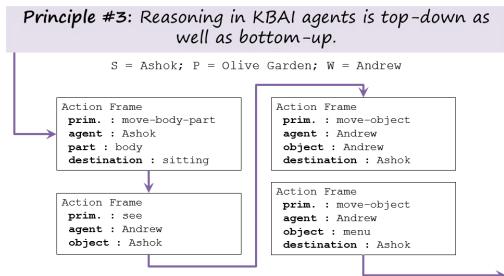


Figure 939: The Third Principle

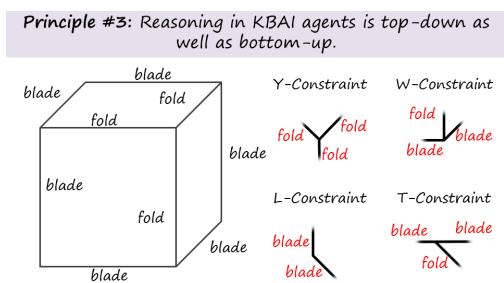


Figure 940: The Third Principle

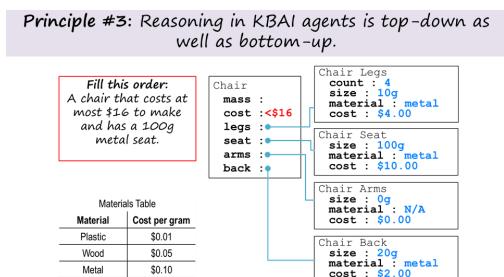


Figure 941: The Third Principle

The third principle of knowledge based AI agents in CS7637 is that reasoning is typically top-down as well as bottom-up. Ordinarily, we assume that data is coming from the world and most of the reasoning is bottom-up as we interpret the data. In contrast to knowledge based AI, low level processing of the data results in the invocation of knowledge structures from memory. These knowledge structures then generate expectations, and the reasoning becomes top down. Frames were an example of this kind of top down reasoning. We had the input, Angela ate lasagna with her dad last night at Olive Garden. And the processing of this input led it to the invocation of this particular frame. Once this

frame has been pulled out of memory, then this particular frame can generate a lot of expectations. For example, was the object alive when Angela ate it? False. Where is the object now? Inside the subject. What is the subject's mood after she had dinner? She was happy. A similar kind of top down generation of expectations occurred when we used frames to understand simple sentences about earthquakes. Scripts are another example of this kind of top-down generation of expectations and expectation based processing. Once we have a script, then that script generates expectations over the next action. It tells us what to look for in the world, even before that happens. And when that doesn't happen, we know an expectation has been violated and we are surprised. Constraint propagation is another way of thinking about top down processing. We will input data in the form of pixels representing this cube. We have to infer that this image, in fact, is that of a cube. We have knowledge about this constraint of various kind of junctions that can occur in the world of cubes and this knowledge then generates expectations of what might be a blade and what might be a fold. This notion of top-down processing using knowledge structures to generate expectations of the world in order to be able to interpret data essential to knowledge based AI. It's also deeply connected with human cognition. Some current theories of human cognition think of brains as predictive machines. We are constantly generating expectations, we are constantly making predictions over the world, that guide our reasoning, that guide our actions.

07 - The Fourth Principle

[Click here to watch the video](#)

Principle #4: KBAI agents match methods to tasks.

Figure 942: The Fourth Principle

Principle #4: KBAI agents match methods to tasks.

Methods
Generate & Test
Means-Ends Analysis
Problem Reduction
Production Systems
Case-Based Reasoning
Planning
Analogical Reasoning

Figure 943: The Fourth Principle

Principle #4: KBAI agents match methods to tasks.

Methods	Tasks
Generate & Test	Configuration
Means-Ends Analysis	Diagnosis
Problem Reduction	Design
Production Systems	Meta-Reasoning
Case-Based Reasoning	Creativity
Planning	Classification
Analogical Reasoning	Systems Thinking

Figure 944: The Fourth Principle

Our fourth principle was a Knowledge Based AI agents match methods to tasks. At the beginning of this course we covered several very powerful problem solving methods like Generate & Test, and Means-Ends Analysis. But because they were very powerful and very general, they also weren't necessarily the best for solving any one problem. We also covered some more specific problem solving methods like planning that addressed a narrower set of problems but addressed those problems very, very well. We also covered several tasks in this class, like configuration and diagnosis and design. These tasks could all be carried out by a variety of methods. For example, we can imagine doing configuration with

Generate & Test or we generate every possible configuration of a certain plan and then test to see which one is best. We could also do configuration by Problem Reduction where we reduce the problem down into the sub parts and solve them individually and then compose them into an overall solution. In this way, knowledge based AI agents match methods to tasks. In some cases we do the matching, we decide that generate and test is the best way to address this diagnosis problem. In other cases we might design AI agents with their own meta-reasoning such that they themselves can decide which method is best for the task that they're facing right now. Note that this distinction between methods and tasks is not always necessarily absolute. Methods can spawn different sub tasks, so for example, if we're doing design by case-based reasoning that spawns new problems to address. And we might address those new problems, those new tasks, with analogical reasoning, or with problem reduction. This gets back to our meta-reasoning notion of strategy integration. In this way, knowledge based AI agents match methods to tasks not only at the top level, but also at every level of the task-subtask hierarchy.

08 - The Fifth Principle

[Click here to watch the video](#)

Principle #5: KBAI agents use heuristics to find solutions that are good enough, though not necessarily optimal.

Figure 945: The Fifth Principle

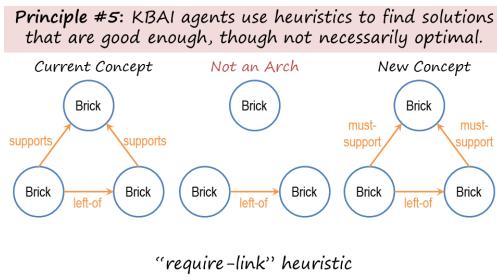


Figure 946: The Fifth Principle

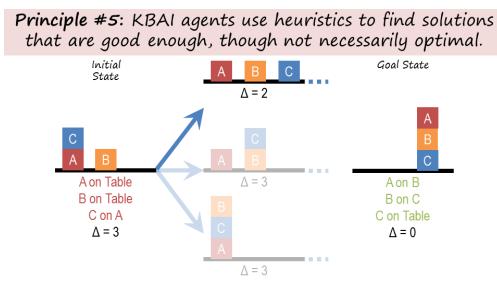


Figure 947: The Fifth Principle

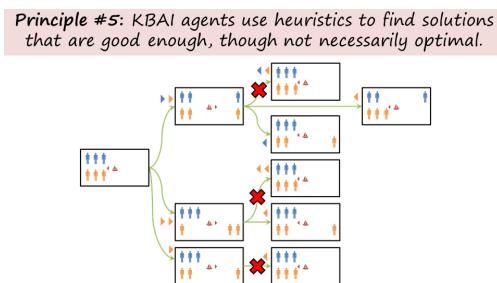


Figure 948: The Fifth Principle

The first principle of knowledge based AI, as we have discussed it in CS7637, is that AI agents use heuristics to find solutions that are good enough, but not necessarily optimal. Some schools of AI put a lot of emphasis on finding the optimal solution to every problem. In knowledge based AI, we consider agents that find solutions that are good enough. Herbert Simon called this *satisficing*. The reason for finding solutions that are only good enough is because of the trade off between computational efficiency on one hand and optimality of solutions on the other. We can find optimal solutions, but that comes with the cost of computational efficiency. Recall one

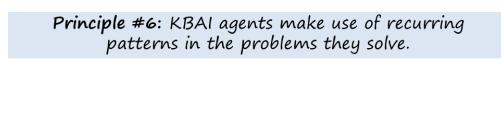
of the conundrums of AI. AI agents are with limited resources, bounded rationality, limited processing power, limited memory size. Yet, most intrusting problems are impractical. How can we get AI agents to solve impractical problems with limited rationality and yet give nearly a ten performance? We can get AI agents to do that if we can focus on finding solutions that are good enough, but not necessarily optimum. Most of the time you and I as human agents do not find optimum solutions. The plan you may have to make dinner for yourself tonight is not necessarily optimum, it's just good enough. The plan that you have to go from your house to your office is not necessary optimum, it's just good enough. The plan that you have to walk from your car to your office is not necessary optimum, it's just good enough. Further, AI agents use heuristics to find solutions that are good enough. They do not do an exhaustive search, even the exhaustive search might yield more optimal solutions because exhaustive search is computationally costly. We came across this notion of heuristic search several times in this course. Once place where we discussed this in some detail, was in incremental concept learning. Given a current concept definition and a negative example, we arrive at a new concept definition by using heuristics like require-link heuristic. The require-link heuristic adds the must clause to this support link between these two bricks. Mean-sense analysis was a heuristic method. It said that given the current position and the goal position, find the differences and then select an operator that will reduce the difference. Because mean-sense analysis was a heuristic method sometimes it ran into problems and did not follow guarantees of optimality. But when it worked, it was very efficient. Another case where we explicitly made use of heuristic laws in the generate interest method. Here we had a heuristic which said, do not generate a state that duplicates a previously generated state which made the method more efficient. Does the focus of knowledge based AI agents is a near real time performance? They're addressing computational intractable problems with bounded resources. And yet being able to solve a very large class of problems in robust

intelligence and flexible intelligence. And that happens not by finding optimal solutions to a narrow class of problems, but by using heuristics to find solutions that are good enough to very large classes of problems. This principle comes very much from theories of human cognition. As I mentioned earlier, humans do not normally find optimal solutions for every problem they face. However, we do manage to find solutions that are good enough, and we do so in near real time, and that's where the power lies

09 - The Sixth Principle

[Click here to watch the video](#)

Principle #6: KBAI agents make use of recurring patterns in the problems they solve.



Principle #6: KBAI agents make use of recurring patterns in the problems they solve.

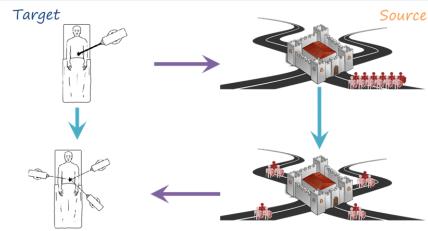


Figure 952: The Sixth Principle

Principle #6: KBAI agents make use of recurring patterns in the problems they solve.

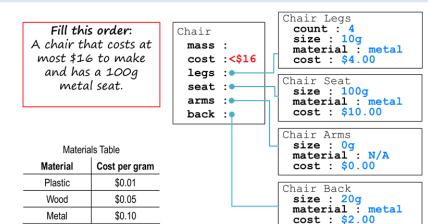


Figure 953: The Sixth Principle

Principle #6: KBAI agents make use of recurring patterns in the problems they solve.

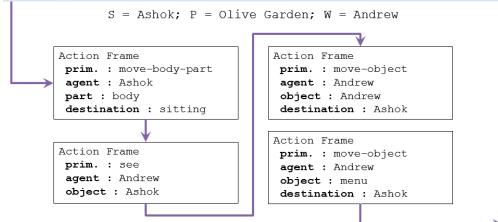


Figure 949: The Sixth Principle

Principle #6: KBAI agents make use of recurring patterns in the problems they solve.

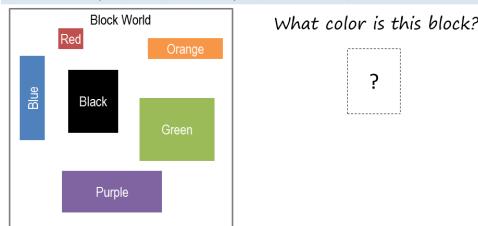


Figure 950: The Sixth Principle

Principle #6: KBAI agents make use of recurring patterns in the problems they solve.



Figure 951: The Sixth Principle

Principle #6: KBAI agents make use of recurring patterns in the problems they solve.

S = Ashok; P = Olive Garden; W = Andrew

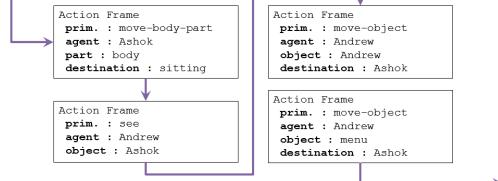


Figure 954: The Sixth Principle

Our sixth principle, was knowledge-based AI agents make use of recurring patterns in the problems that they solve. These agents are likely to see similar problems over and over again and make use of the underlying patterns behind these similar problems to solve them more easily. We talked about this first with learning about recording cases. Here we assumed that we had a library of cases, and that the solution to a former case would be the exact solution to a new problem. Ashok's example of tying shoe laces was similar to this. When we tie our shoelaces, we aren't resolving the problem of tying our shoelaces from scratch. Instead we're just taking the solution from an earlier

time when we tied our shoelaces and doing it again. We assumed that the solution to the old problem will solve this new similar problem. In case-based reasoning, however, we talked about how the exact solution to an old problem won't always solve new problems. Instead sometimes we have to adapt an old problem. Here we assumed that there were recurring patterns in the world that would help us solve these new and novel problems based on previous experiences. Even though the new experience is novel, the pattern is similar to a prior experience. Analogical reasoning is very deeply rooted in this principle. Here we explicitly talked about the idea of taking patterns from one problem, abstracting them, and transferring them to a problem in a different domain. Whereas in case-based reasoning, the pattern was within a domain, here the pattern can span different domains. In configuration, we assumed that the underlying design, the underlying plan for a certain device or product was pretty similar each time. But there were certain variables that had to be defined for an individual instance of that object. In a chair example, the overall design of a chair is a recurring problem, they all have legs, they all have seats, they all have backs, but the individual details of a specific chair might differ. Now, it might be tempting to think that this is actually at odds with the previous principal when the knowledge-based AI agent's consult a novel problems. Here we're saying that knowledge-based AI agents solve recurring problems based on recurring patterns, but in fact these are not mutually exclusive. Knowledge-based AI agents leverage recurring patterns in the world, but they do so in conjunction with the other reasoning methods to allow them to also address novel problems.

Principle #7: The architecture of KBAI agents enables reasoning, learning, and memory to support and constrain each other.

Figure 955: The Seventh Principle

Principle #7: The architecture of KBAI agents enables reasoning, learning, and memory to support and constrain each other.

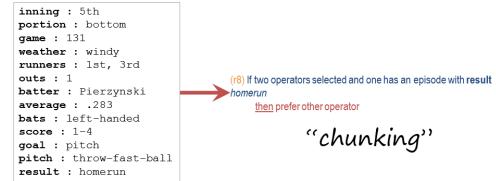


Figure 956: The Seventh Principle

Principle #7: The architecture of KBAI agents enables reasoning, learning, and memory to support and constrain each other.

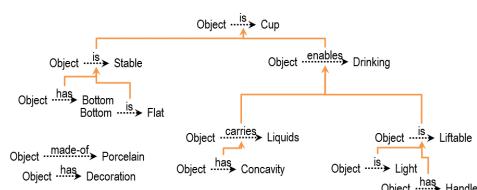
If an animal flies and is not a bird, it is a bat.

$$\text{Flies}(\text{animal}) \wedge \neg\text{Bird}(\text{animal})$$

If $\text{Flies}(\text{animal}) \wedge \neg\text{Bird}(\text{animal})$:
Then $\text{Bat}(\text{Animal})$

Figure 957: The Seventh Principle

Principle #7: The architecture of KBAI agents enables reasoning, learning, and memory to support and constrain each other.



10 - The Seventh Principle
[Click here to watch the video](#)

Figure 958: The Seventh Principle

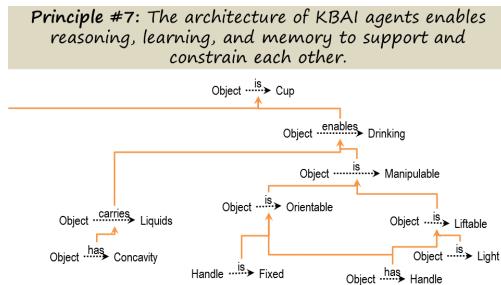


Figure 959: The Seventh Principle

So the seventh and last principle of knowledge based AI agents in CS7637 is that the architecture of knowledge based AI agents enables reasoning, learning, and memory to support and constrain each other. Instead of building a theory of reasoning or problem solving by itself or a theory of learning by itself or a theory of memory by itself, we are trying to build unified theories, theories where reasoning, learning, and memory coexist. Memory stores and organizes knowledge. Learning acquires knowledge. Reasoning uses knowledge. Knowledge is the glue between these three. One place that we saw reasoning, learning, and memory coming together very well was in production systems. When reasoning failed, an impasse was reached then. Memory provided some episodic knowledge and a learning mechanism of chunking extracted a rule from that episodic knowledge. And that rule broke the impasse and reasoning could proceed a pace. This is a clear example where reasoning, learning, and memory came together in a unified architecture. In logic, memory, or the knowledge base, may begin with a set of axioms. And those set of axioms decide what we can prove using that particular logic. To look at the problem, conversely, depending upon the reasoning, we need to put into the knowledge base, so that the reasoning can be supported. Exploration of this learning was under the place. Where reasoning learning and memory came together so well. Memory supplied us with the earlier precedents. Reasoning led to the composition of this explanation which explained why this instance was an example of a cup. This lead to learning about the connections between this weirdest precedence to an explanation. Here in the correcting mistakes

was yet another example of, learning, reasoning, and memory coming together. Where a failure occurred, when the agent used it's previous knowledge, memory, to reason and identify the fault responsible for the failure, reasoning, and then corrected that particular fault, learning, in order to get the correct model. This knowledge based paradigm says that we want to be able to reach unified, that connect reasoning, learning and memory. And this also connects very well with human cognition, human cognition, of course, has reasoning and learning and memory intertwined together. It is not as if memory and human cognition works by itself or learning works by itself or reasoning works by itself. You cannot divorce them from each other.

11 - Current Research

[Click here to watch the video](#)

CALO:
“Cognitive Assistant that Learns and Organizes”

Cyc and OMCS:
Knowledgebases of everyday common sense knowledge

Wolfram Alpha:
A computational knowledge and answer engine.

VITA:
A computational model of visual thinking in autism, based on RPM.

Dramatis:
A computational model of suspense and drama in stories.

DANE:
Support for design based on analogies to natural systems.

Figure 960: Current Research

Knowledge based is a dominate field with a very active research program. There are number of exiting projects going on right now. Here is a small list of them. CALO is a project, in which cognitive assistant learns and organizes knowledge. CALO, in fact was a pick cursor for the CD program of Apple. Cyc and OMCS. OMCS stands for Open Mind Common Sense. Cyc and OMCS are two large knowledge bases to support everyday common sense of reasoning. Wolfram Alpha is a new kind of search engine that uses some of the same kind of pilot structures we have considered in this particular class, different from many of the search engines. The three projects in the right column are projects here at Georgia Tech. VITA is a computational model of visual thinking in autism. And particular it solves

problems in the real world raven's progressive matrices test using only visual spacial representations. Dramatis is a computational model suspense in drama and stories. Recall that in this class we talked about the theory of humor and surprise. Dramatis tries to do the same thing for suspense. DANE is a system for supporting design based on analogies to natural systems. We've come across this idea of biologically inspired design earlier in the class and DANE supports that kind of biologically inspired design. We have provided references for these and many other knowledge based AI projects in the class notes. Your welcome to explore depending on your inquests.

12 - Our Reflections

[Click here to watch the video](#)

This brings us to the end of the course. There are many more topics that we could talk about, and we encourage you to look at the additional readings for the course. As we said at the beginning, we have had a lot of fun putting this course together, and we have learned a lot, too. We hope you have enjoyed the course as well. We're also really eager to get your feedback. What did you enjoy about this course? And what could have been improved? What surprised you about this course, either in the content or in the way that it was administered? Are there things that you've seen other courses do that you wish we'd done, or are things that we did that you'd like to see other courses do? We are very interested in your answers to these questions, so please feel free to contact either one of us with your answers, and fill out the course survey as well.

We'd also like to thank several people for helping make this course possible. First, our video editor, Aaron, who's done a phenomenal job making these videos look so good and frankly making us look presentable. We'd also like to thank the rest of you at Assety, especially Jenny Kim, Jason Barrows, and Katie Reicuit for providing a great infrastructure for course development. We'd also like thank our colleagues here at Georgia Tech, including David White at the college of computing. And Mark Weston and his staff at the Georgia Professional Education Department. It's been a real fun journey for us. We expect to hear from you. We hope it's the beginning of a beautiful friendship.

Summary

This module covers all the topics of the course. Hope you enjoyed this KBAI ebook. Let the authors know if you have any comments or suggestions for improving it.

References

1. Winston, P., Artificial Intelligence: Videos and Material [Click here](#)
2. Russell, S., & Norvig, P. Artificial Intelligence: A Modern Approach.
3. Stefk, M. Introduction to Knowledge Systems.
4. aitopics.org [Click here](#)
5. Bostrom Nick, Yudkowsky Eliezer, The Ethics of Artificial Intelligence.

Exercises

None.