Artificial Intelligence

Algorithms and Applications with Python

Chapter 05

Dr. Dominik Jung
dominik.jung42@gmail.com

</> course material: ↗ https://github.com/dominikjung42

Outline

5 Knowledge Reasoning - Fundamental Algorithms and Concepts

Image source: ↗ Pixabay (2019) / ↗ CC0

► **What we will learn:**

- We will design logical agents that can form representations of the world
- For that purpose we will formalize knowledge and reasoning processes with representation languages like propositional or first-order logic
- Use inference processes to derive new representations about the world, and use these new representations for decision-making and planning

► **Duration:**

- 180 min

► **Relevant for Exam:**

- 5.1 – 5.2

5.1 Problem: Search-Approach Can Not Handle Many Real-World Problems

- In the previous chapter build intelligent agents by modelling problems as search problems, and using clever search algorithms to solve these problems
- **Problem**: Search algorithms generate graphs or trees with successors and evaluate them, but do not "understand" anything

5.1 Motivation of Knowledge-Reasoning in Artificial Intelligence

- In the past we tried to create intelligent agents by modelling problems as search problems, and using clever search algorithms

- However, as we have discussed, most real-world problems do not have well-definied solutions and hence our „search-approach“ will not work for these kind of problems human can solve…

- What do humans do?
  → Humans know things and use it to act.

- Or, as AI researchers say, they have **real-world knowledge** and **do reasoning** to solve real-world problems

Adapted from Beierle, C., & Kern-Isberner, G. (2014); Rusell, S., & Norvig, P. (2016) | Quote and Image source: Tyrion Lannister, Game of Thrones, Season 6, Episode 2

5.1 New Approach: Using Knowledge to Create Artificial Intelligence

- Knowledge and reasoning is important when dealing with partially observable environments
- Knowledge-based agents and systems benefit from knowledge expressed in general forms combining in there inference process
- **Our new approach**: Find a way to represent knowledge and implement inferences procedures to derive solutions in our agents

D **Knowledge-based System**

An expert system or knowledge-based system is one that solves problems by applying knowledge that has been garnered from one or more experts in a field (Norvig, 1992)

5.1 Example: Credit Scoring System

? How can you / an AI conclude if a customer gets a credit or not?

Scenario

You work in a big car company and are asked to build a credit scoring agent to automate most of the car credits. Consider the following aspects:

- Customers have to register succesfully with their credit card and correct pin when they want to buy the car with a financial leasing contract.
- Furthermore, the customer has to pay at least the minimum montly rate
- If he has no other open car credits, the contract is accepted, otherweise there has to be a manual check of the financial situation

Variable	Values
Credit Card	{accepted, not accepted}
PIN	{true, false}
Montly credit rate	{≤ minimum amount, > minimum amount}
Other open credits	{true, false}
Accept	{yes, no}
Manual check	{ok, not ok}

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014) | Image source: ↗ Pixabay (2019) / ↗ CC0

5.1 Implement Credit Reasoning

- The central point of a knowledge-based system is to represent the problem-related knowledge as directly as possible
- For that purpose we have to keep the actual processing separate from knowledge representation
- One very intuitive approach for such a direct knowledge representation is the use of rules.

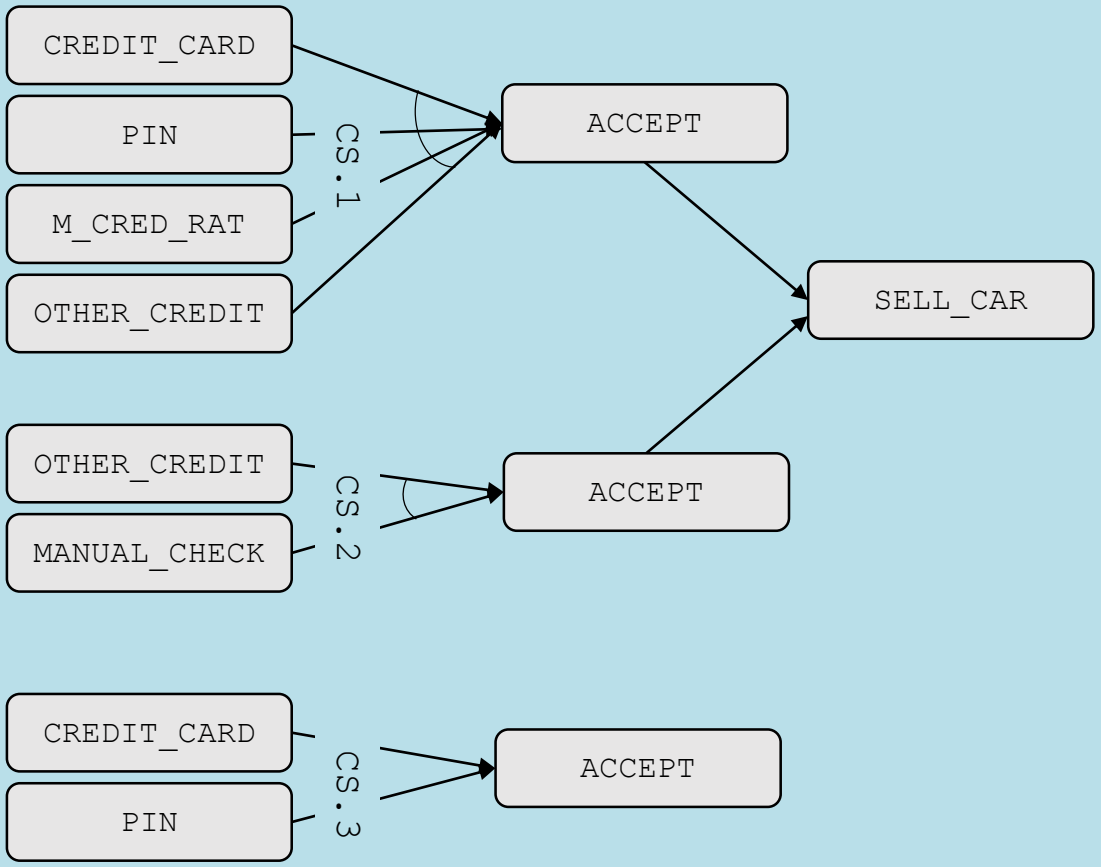
D A rule is a logical statement that relates two or more objects and includes two parts, the premise and the conclusion (Castillo, E et al. 2012, p.23)

```
[CS.1]
if
    CREDIT_CARD = accepted and
    PIN = true and
    MONTLY_CREDIT_RATE > MIN_AMOUNT and
    OTHER_CREDITS = false

then
    ACCEPT = yes
```

```
[CS.2]
if
    OTHER_CREDITS = true and
    MANUAL_CHECK = not_ok

then
    ACCEPT = no
```

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014)

5.1 Rule Networks

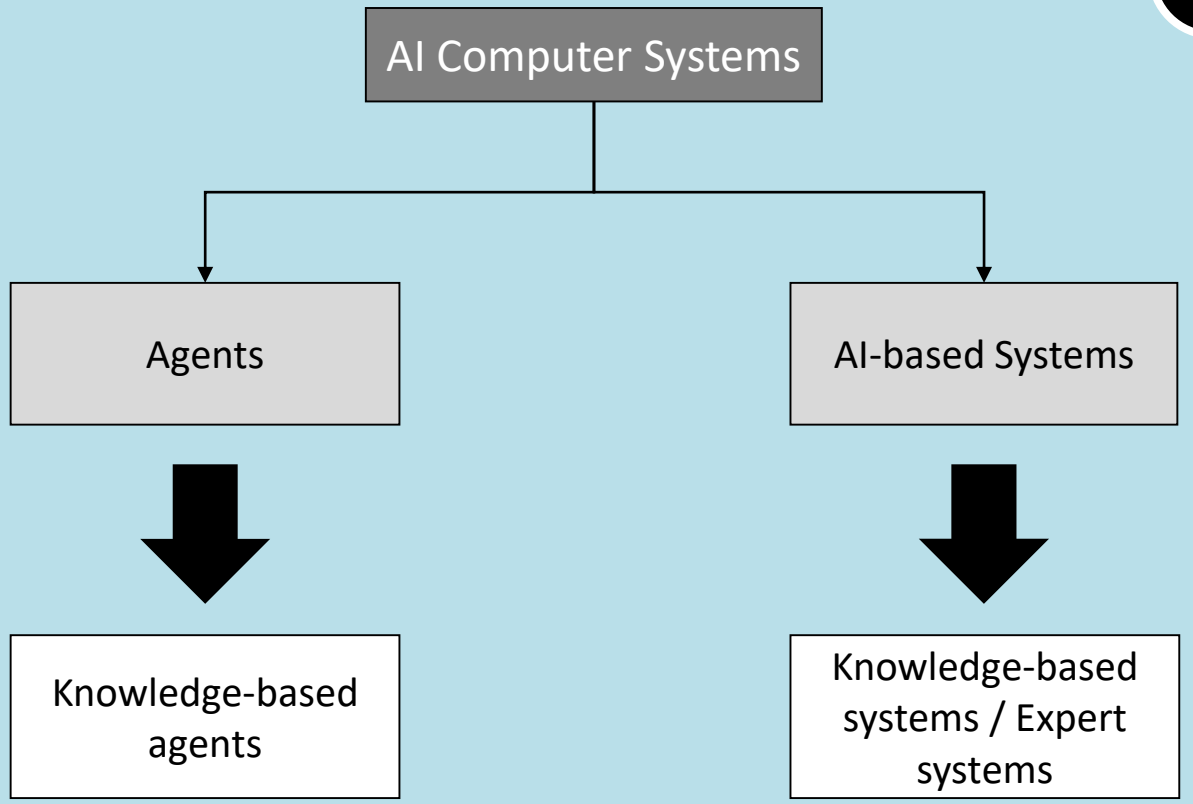



```
[CS.1]
if
    CREDIT_CARD = accepted and
    PIN = true and
    MONTLY_CREDIT_RATE > MIN_AMOUNT and
    OTHER_CREDITS = false
then
    ACCEPT = yes
```

```
[CS.2]
if
    OTHER_CREDITS = true and
    MANUAL_CHECK = not_ok
then
    ACCEPT = no
```

```
[CS.3]
if
    CARD_DECLINED = true or
    SCHUFA_WARNING = true
then
    ACCEPT = no
```

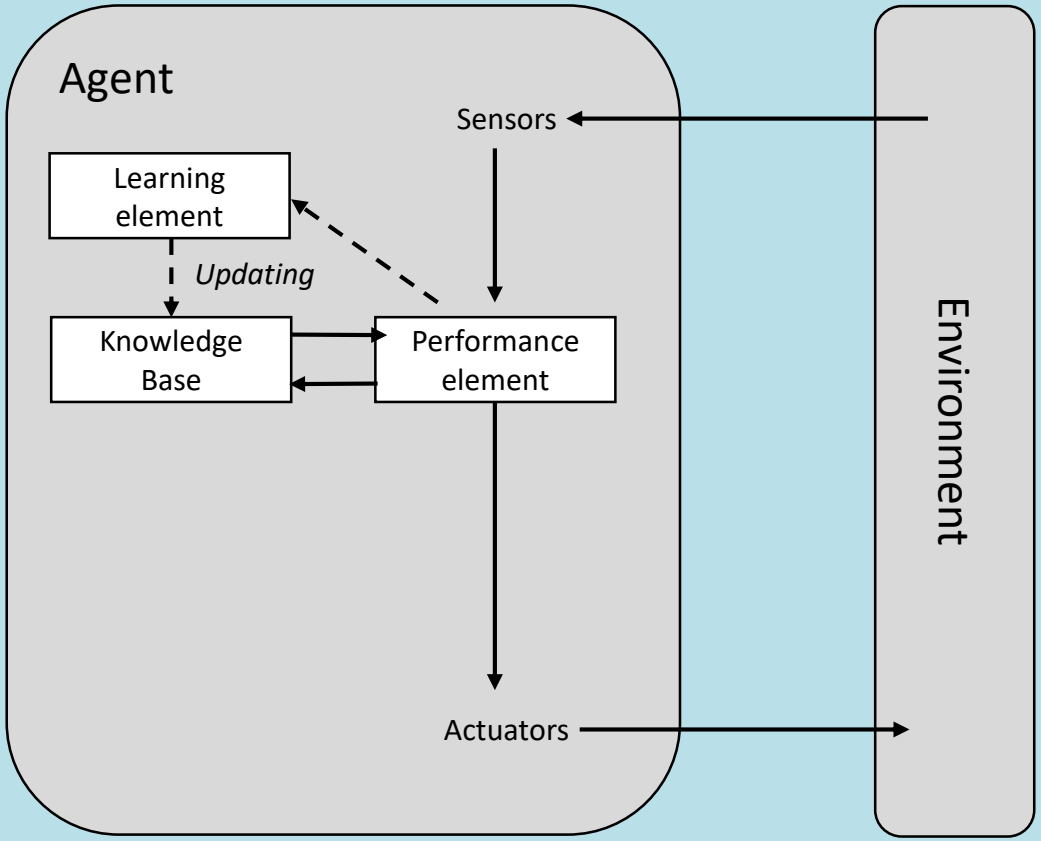
5.1 Use Cases of Knowledge Reasoning



Go back to lecture 1 to learn more about these two types of computer systems

5.1 Knowledge-based Agent

▶ Adds percepts and action's results to knowledge base, and derives actions from the knowledge base



Adapted from Rusell, S., & Norvig, P. (2016)

Inference Procedure

1. Agent TELLs the knowledge base what it perceives.
2. Agent ASKs the knowledge base what action it should perform
3. The Agent TELLs the knowledge base which action was chosen, and then executes the action.

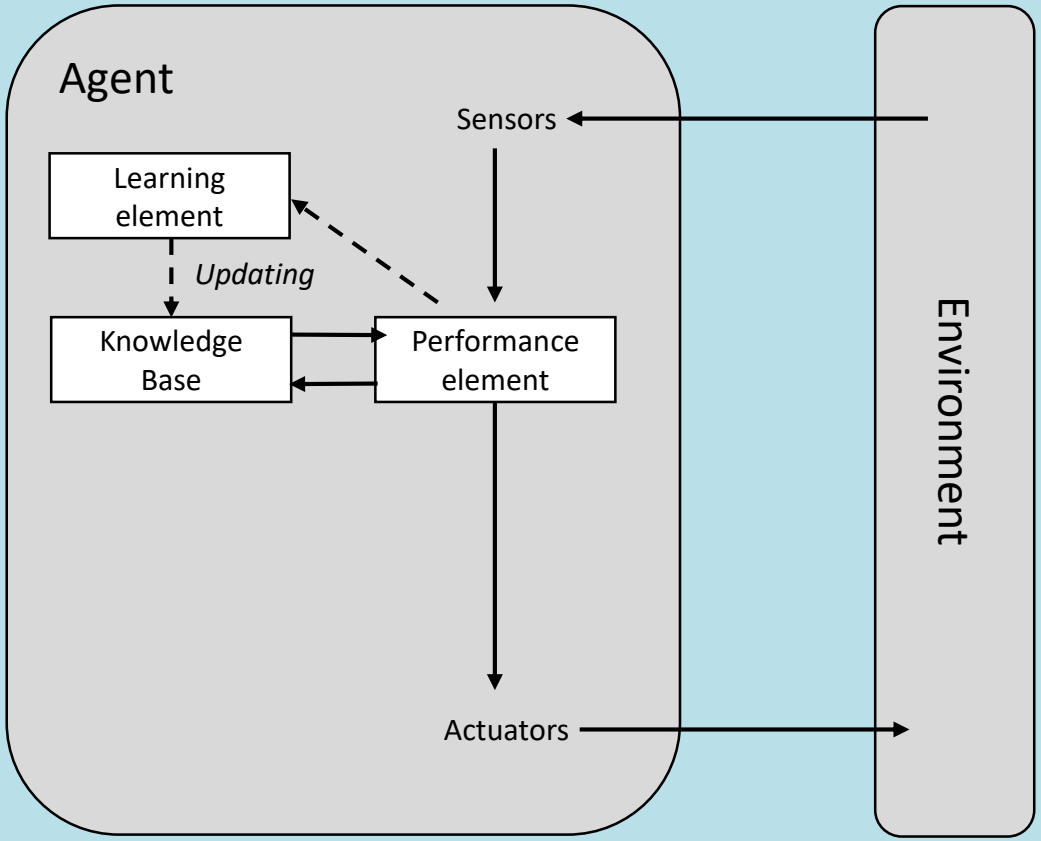
Knowledge Base

Knowledge is represented in the knowledge base

- Case-Specific knowledge (temporary)
- Rule-based knowledge
  - Domain knowledge
  - General knowledge

5.1 Knowledge-based Agent

▶ Adds percepts and action's results to knowledge base, and derives actions from the knowledge base



Algorithm: Knowledge-based Agent

persistent:
KB, *a knowledge base* t,
a counter, initially 0, *indicating time*

$TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))$

$action \leftarrow ASK(KB, MAKE-ACTION-QUERY(t))$

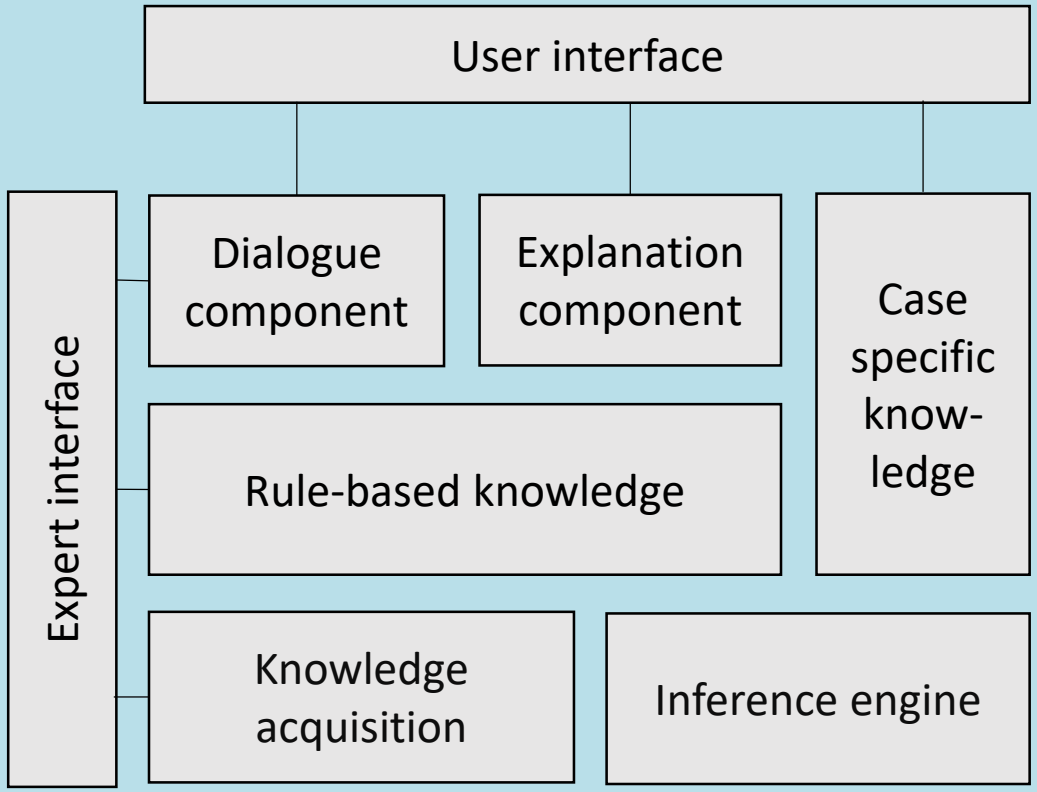
$TELL(KB, MAKE-ACTION-SENTENCE(action, t))$

$t \leftarrow t + 1$

return *action*

Adapted from Rusell, S., & Norvig, P. (2016)

5.1 Architecture of Knowledge-based System (KBS)

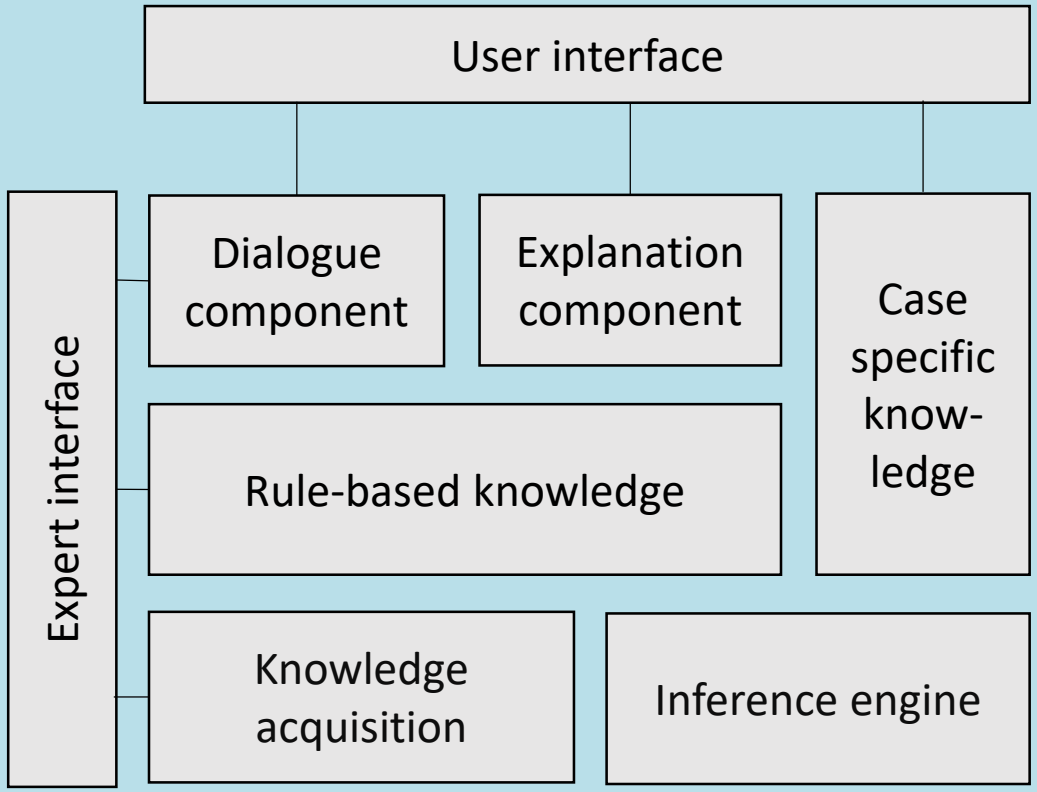


Adapted from Beierle, C., & Kern-Isberner, G. (2014)

From Kowledge to Expert Knowledge

- We assume, that experts have above-average abilities to solve problems in a specific field
- Experts can solve problems using incomplete and uncertain Solving knowledge
- Experts use heuristic knowledge to solve specific problems and exploit their experience
- Experts are rare and expensive, and their knowledge can be lost if they are lost

5.1 Architecture of Knowledge-based System (KBS)



Adapted from Beierle, C., & Kern-Isberner, G. (2014)

Inference Engine

- An inference engine tries to dervice answers from a knowledeg base
- It is the „brain“ of the expert system that provides a methodology for reasoning about the information in the knowledge base, and for formulating conclusions

Interfaces

- **Expert interface**: Add and store new expert knowledge in the knowledge base, revise existing knowledge
- **User interface**: Specify situation and ask for expert advice

5.1 Example: Popular Knowledge-based Systems

- **Dendral**: Pioneering work developed in 1965 for NASA at Standford University
- **Drilling Advisor**: Developed in 1983 by Teknowledge for oil companies to replace human drilling advisors
- **Mycin**: Developed in 1970 at Stanford by Shortcliffe to assist internists in diagnosis and treatment of infectious diseases
- **Xcon/RI**: Developed in 1978 to assist the ordering of computer systems by automatically selecting the system components based on customer‘s requirements

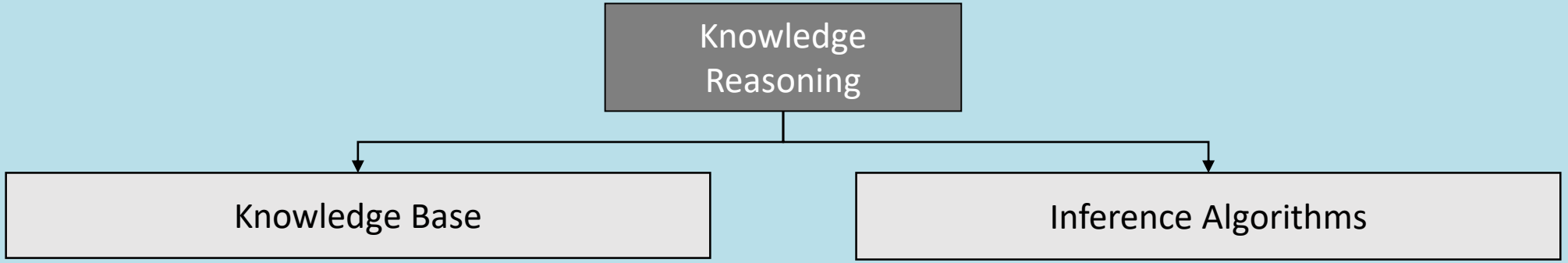
Image source: ↗ A Symbolics Lisp Machine (2019) by Michael L. Umbricht and Carl R. Friend (Retro-Computing Society of RI) / ↗ CC BY-SA 3.0

Adapted from Beierle, C., & Kern-Isberner, G. (2014)

5.1 Knowledge Reasoning as Researchfield in Artificial Intelligence

D **Knowledge Reasoning**

Knowledge Representation is the study of how “what we know” can at the same time be represented as comprehensibly as possible and reasoned with as effectively as possibly from an information system (based on Brachman & Levesque, 1992).

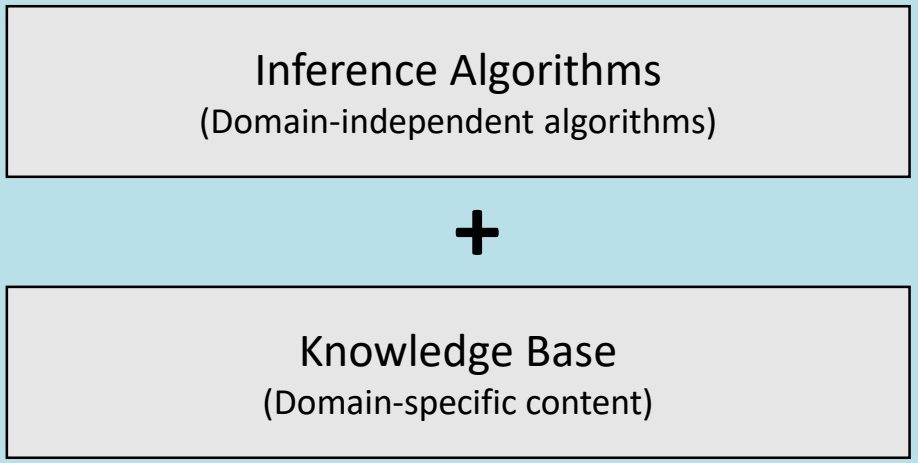


Seperation of concerns:

- clear distinction of problem description and problem solution
- knowledge of domain can be expressed directly

Adapted from Beierle, C., & Kern-Isberner, G. (2014); Rusell, S., & Norvig, P. (2016)

5.1 Using Knowledge Reasoning in Artificial Intelligence



- Knowledge base (KB) = set of sentences in a formal language
- There are mainly two approaches to implement the agent's logic (or other type of AI system):
  - Declarative approach
  - Procedural approach
- Distinction between data and program

D **Knowledge Base**

The component of an expert system that contains the system's knowledge organized in collection of sentences about the system's domain.

Adapted from Beierle, C., & Kern-Isberner, G. (2014); Rusell, S., & Norvig, P. (2016)

5.1 Various Levels of Knowledge-based Agents and Systems

- **Knowledge level**: What the agent knows, and the agent's goals.
- **Logical level**: how the knowledge representation of knowledge is stored. At this level, sentences can be encoded into different logics
- **Implementation level**: physical representation of logic and knowledge. At the implementation level agent perform actions as per logical and knowledge level.

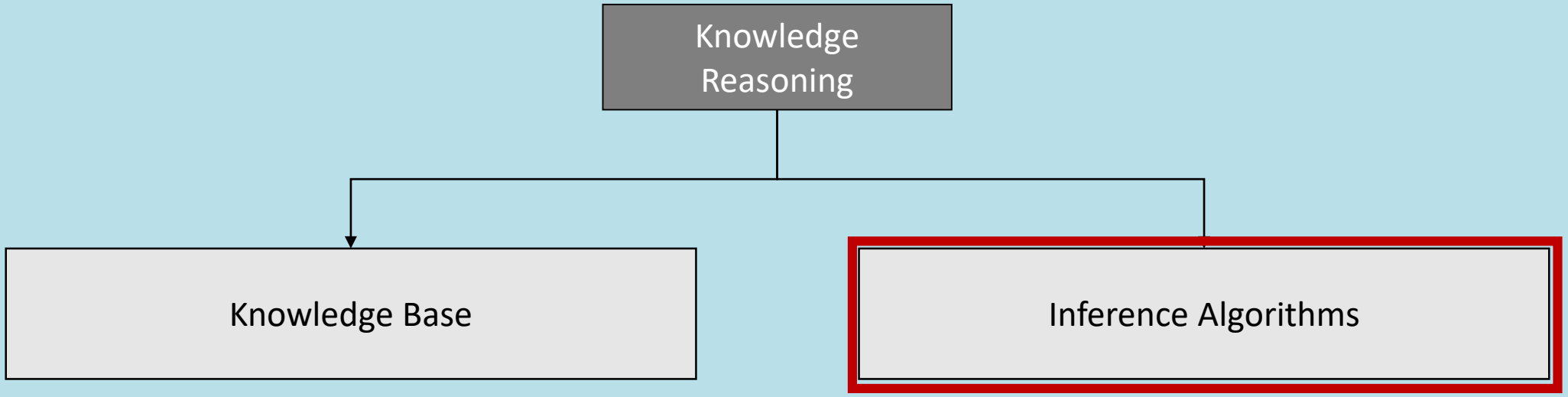
5.1 Todo: Knowledge-Based Agent Requirements

- To build knowledge-based agents we have to specify the knowledge (what an agent has to know), and the agent logic (how knowledge is processed in the agent)

- For that purpose:
  - An agent should be able to represent states, actions, etc.
  - An agent should be able to incorporate new percepts, and to update internal representations of the world
  - An agent can deduce hidden properties of the world, and deduce appropriate actions

Adapted from Beierle, C., & Kern-Isberner, G. (2014); Rusell, S., & Norvig, P. (2016)

5.2 Inference

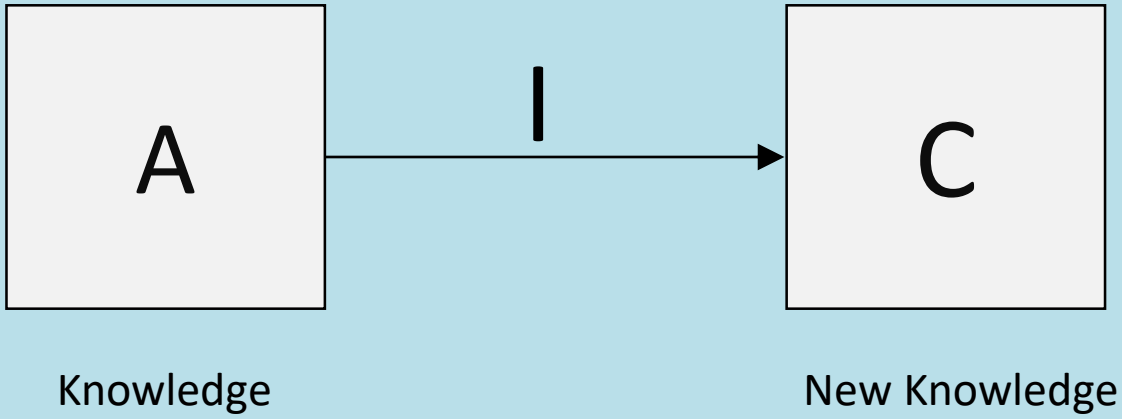


Define formal system for handling facts so that in each case where the agent draws conclusion from the available information, that conclusion is guaranteed to be correct if the available information is correct.

Adapted from Rusell, S., & Norvig, P. (2016)

5.2 What is Reasoning?

- A reasonig process usally consists of two parts: antecedant, inference rule, and conclusion
- The objective of reasoning is to find the missing component

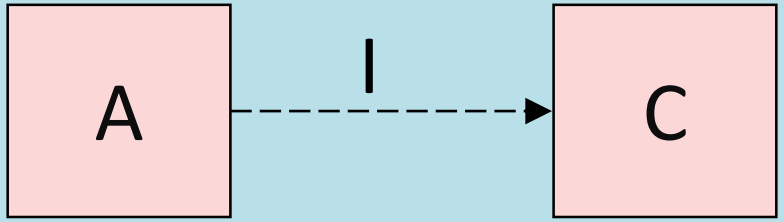


- Conclusion is missing: Deduction
- Inference is missing: Induction
- Antecedant is missing: Abduction

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014); Peirce, C (1931, 103 ff.)

5.2 Induction

Visualization



Characteristics

- Given a set of D of basic examples. The hypothesis H follows inductively from D and background knowledge B
- <=> $B \cup H \rightarrow D, B \not\rightarrow D, B \cup D \not\rightarrow \neg H.$
- **Application**: Data Mining, Economy

Case: These beans are [randomly selected] from this bag.

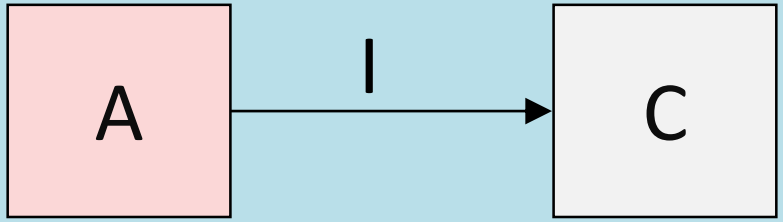
Result: These beans are white.

Rule: All the beans from this bag are white.

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014); Peirce, C (1931, 103 ff.)

5.2 Deduction

Visualization



Characteristics

- From a set of formulas F follows B
  <=> There is a sequence of rules to derive B
- $\frac{F, F \rightarrow B}{B}$.
- **Application**: Mathematics

Rule: All the beans from this bag are white.

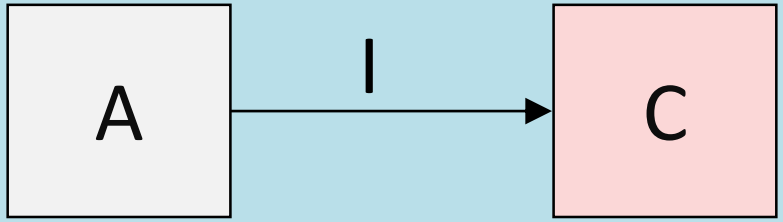
Case: These beans are from this bag.

Result: These beans are white.

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014); Peirce, C (1931, 103 ff.)

5.2 Abduction

Visualization



Characteristics

- H follows from background knowledge B and observations D
- abductive <=> $B \cup H \rightarrow D$
- **Application**: Medical Diagnosis, Car Repairing, Failure Explanation

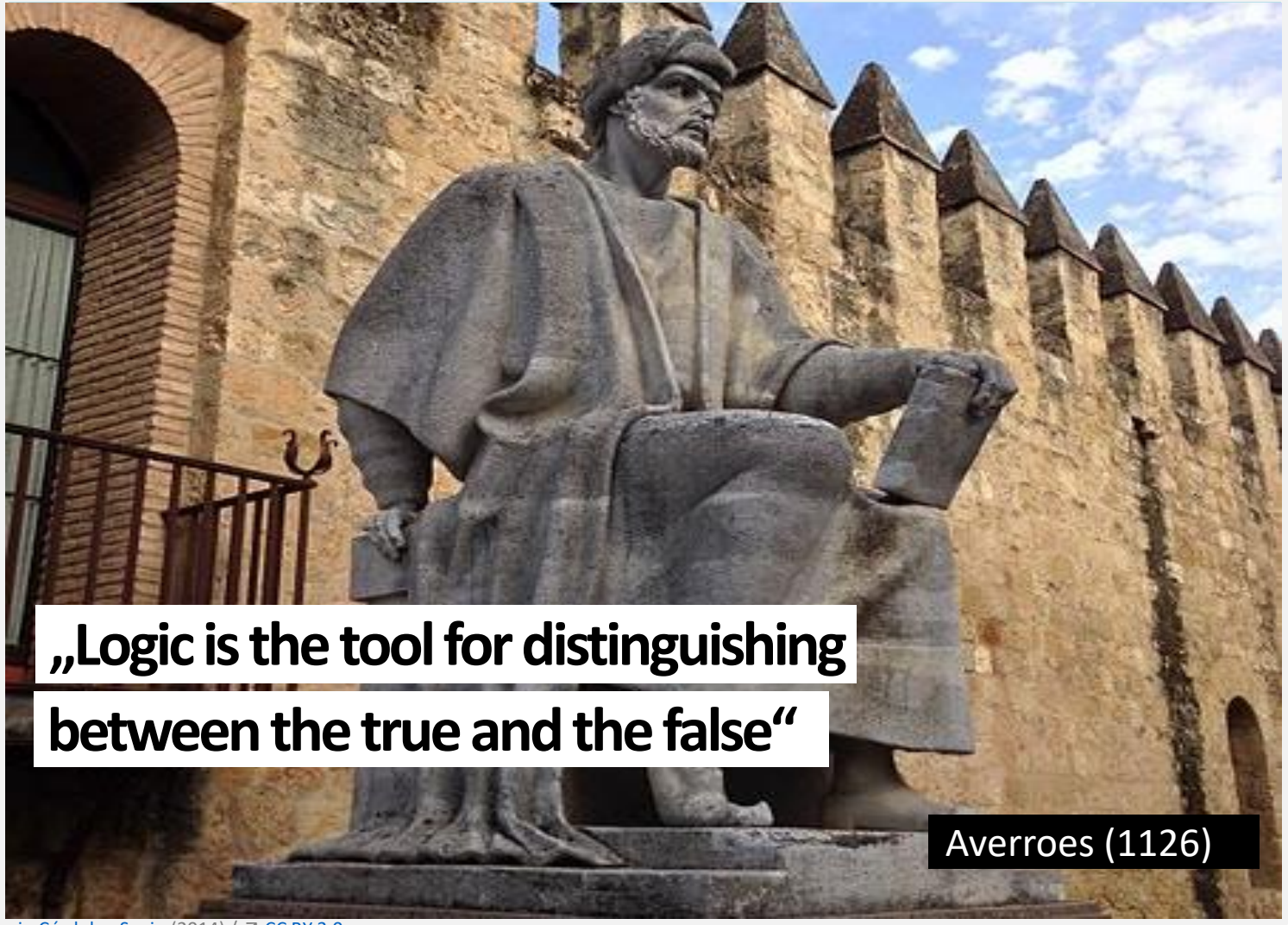
Rule: All the beans from this bag are white.

Result: These beans [oddly] are white
.
Case: These beans are from this bag.

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014); Peirce, C (1931, 103 ff.)

5.2 Logic and Formalization of Reasoning

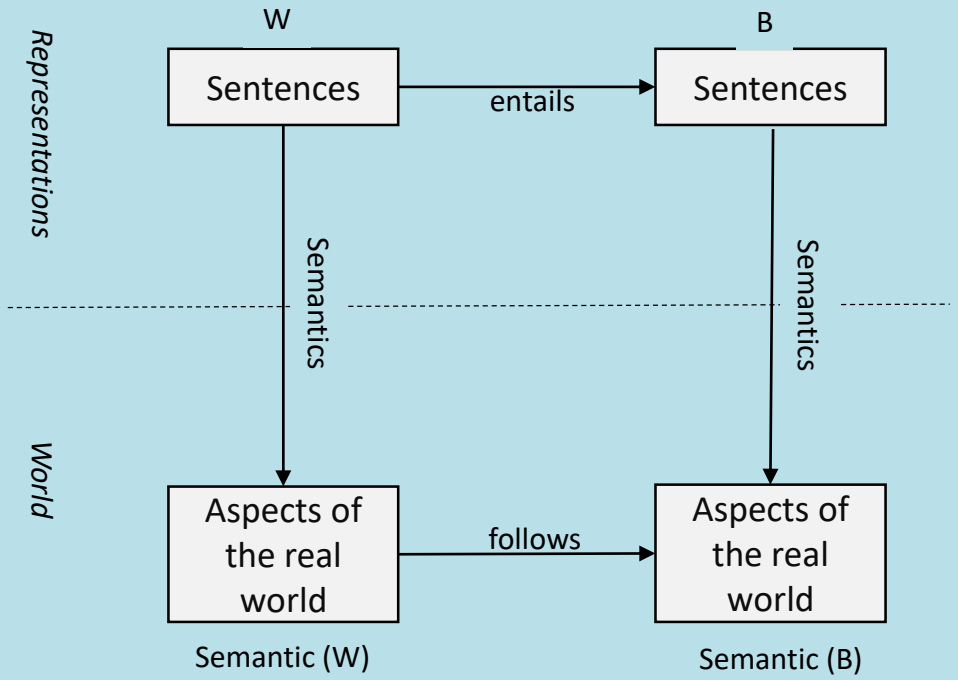
- **We need rules** and systematics for manipulating facts **to allow computational reasoning**
- **Logic** is such a formal system for **handling facts so that true conclusions may be drawn**
- We can apply logical rules to our knowledge base to deduce new information



Adapted from Rusell, S., & Norvig, P. (2016) | Image source: ↗ Statue of Averroes in Córdoba, Spain (2014) / ↗ CC BY 3.0

5.2 What is a Logic in Knowledge Reasoning?

- **Syntax:** rules for constructing valid sentences
  - e.g., $x + 2 = y$ is a valid arithmetic sentence
- **Semantics:** "meaning" of sentences, or relationship between logical sentences and the real world
  - Specifically, semantics defines truth of sentences
  - e.g., $x + 2 = y$ is true in a world (≈model) where $x = 5$ and $y = 7$



Adapted from Beierle, C., & Kern-Isberner, G. (2014; Rusell, S., & Norvig, P. (2016) | Image source: Rusell, S., & Norvig, P. (2016)

5.2 Syntax of Propositional Logic

- **Atomic sentence (literal):**
  - A *proposition symbol* representing a true or false statement

- **Negation:** If **A** is a sentence, $\neg$**A** is a sentence
- **Conjunction:** If **A** and **B** are sentences, **A** $\wedge$ **B** is a sentence
- **Disjunction:** If **A** and **B** are sentences, **A** $\vee$ **B** is a sentence
- **Implication:** If **A** and **B** are sentences, **A** $\Rightarrow$ **B** is a sentence
- **Biconditional:** If **A** and **B** are sentences, **A** $\Leftrightarrow$ **B** is a sentence

The related symbols $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ are called *logical connectives*

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014)

5.2 Semantics

- The semantic defines how the truth of a sentence is computed.

- For that, propositional logic has the following rules:

Given a model where…

A is true, $\neg$**A** is false
A and B is true, **A** $\wedge$ **B** is true
A or B are true, **A** $\vee$ **B** is true
A is **true** and **B** is false, **A** $\Rightarrow$ **B** is false
A and B are both true or false, **A** $\Leftrightarrow$ **B** is true

Adapted from Rusell, S., & Norvig, P. (2016)

5.2 Semantics and Truth Tables

- The semantic defines how the truth of a sentence is computed.
- A truth table specifies the truth value of a composite sentence for each possible assignments of truth values to its atoms

A	B	¬A	A ∧ B	A ∨ B	A => B
false	false	true	false	false	true
false	true	true	false	true	true
true	false	false	false	true	false
true	true	false	true	true	true

- The truth value of a more complex sentence can be evaluated *recursively* or *compositionally*

Adapted from Rusell, S., & Norvig, P. (2016)

5.2 Logical Equivalences

- Two sentences are logically equivalent if they have the same truth value for every setting of their propositional variables

A	B		$A \vee B$	$\neg(\neg A \wedge \neg B)$
false	false		false	false
false	true		true	true
true	false		true	true
true	true		true	true

- `A OR B` and `NOT(¬A ∧ ¬(B))` are logically equivalent; Tautology = logically equivalent to True

Adapted from Rusell, S., & Norvig, P. (2016)

5.2 Logical Equivalences

D

Commutatitvity	$(a\ OR\ b)\ \equiv\ (b\ OR\ a)$
	$(a\ AND\ b)\ \equiv\ (b\ AND\ a)$
Associativity	$((a\ AND\ b)\ AND\ c)\ \equiv\ (a\ AND\ (b\ AND\ c))$
	$((a\ OR\ b)\ OR\ c)\ \equiv\ (a\ OR\ (b\ OR\ c))$
Double-negation elimination	$NOT(NOT(a))\ \equiv\ a$
Contraposition	$(a\ =>\ b)\ \equiv\ (NOT(b)\ =>\ NOT(a))$
Implication elimination	$(a\ =>\ b)\ \equiv\ (NOT(a)\ OR\ b)$
De Morgan	$NOT(a\ AND\ b)\ \equiv\ (NOT(a)\ OR\ NOT(b))$
	$NOT(a\ OR\ b)\ \equiv\ (NOT(a)\ AND\ NOT(b))$
Distributitivity	$(a\ AND\ (b\ OR\ c))\ \equiv\ ((a\ AND\ b)\ OR\ (a\ AND\ c))$
	$(a\ OR\ (b\ AND\ c))\ \equiv\ ((a\ OR\ b)\ AND\ (a\ OR\ c))$

Adapted from Rusell, S., & Norvig, P. (2016)

5.2 Entailment

- **Entailment** means that a sentence follows from the premises contained in the knowledge base (or a model):

$$KB \models \alpha$$

- Knowledge base *KB* entails sentence α if and only if α is true in all models where *KB* is true
  - e.g., x + y = 4 entails 4 = x + y

5.2 From Entailment to Logical Inference

- **Logical inference algorithm:** a procedure for generating sentences that follow from a knowledge base KB

- Two central characteristics: soundness and completeness

D **Soundness**

Inference algorithm derives true conclusions given true premises

D **Completeness**

Inference algorithm derives all true conclusions from a set of premises

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014)

5.2 Inference and Model Checking

- To make implement intelligent behavior our agent has to check whether a sentence α is entailed by its KB or model
- For that we could enumerate all possible models of the KB (truth assignments of all its symbols), and check that α is true in every model in which KB is true
  - Is this sound?
  - Is this complete?

- **Problem**: if KB contains n symbols, the truth table will be of size 2^n

- **Better idea**: use *inference rules*, or sound procedures to generate new sentences or *conclusions* given the *premises* in the KB

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014)

5.2 Reasoning patterns/ Inference rules

- Modus Ponens

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$

$\alpha \Rightarrow \beta, \alpha$ ← premises

β ← conclusion

- And-elimination

$$\frac{\alpha \wedge \beta}{\alpha}$$

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014)

5.2 Reasoning patterns/ Inference rules

And-introduction

$$\frac{\alpha,\ \beta}{\alpha \wedge \beta}$$

Double negative elimination

$$\frac{\neg\neg\alpha}{\alpha}$$

Or-introduction

$$\frac{\alpha}{\alpha \vee \beta}$$

Unit resolution

$$\frac{\alpha \vee \beta, \neg\beta}{\alpha}$$

5.2 Resolution

- Furthermore, we need to prove that our inference procedures are complete

- To prove a single sentence, e.g. $KB \models \alpha$, assume $KB \wedge \neg \alpha$ and derive a contradiction

$$\frac{\alpha \vee \beta, \neg\beta \vee \gamma}{\alpha \vee \gamma} \quad \text{or} \quad \frac{\alpha \vee \beta, \beta \Rightarrow \gamma}{\alpha \vee \gamma}$$

- For that purpose, we have to rewrite $KB \wedge \neg \alpha$ as a disjunctions of literals or conjunction of clauses (CNF)

Adapted from Castillo, E. et al. (2012); Rusell, S., & Norvig, P. (2016)

5.2 Resolution Algorithm

Algorithm: PL-RESULUTION(KB,α)

Inputs: *KB, a knowledge base*
α sentence of propositional logic

clauses ← *set off clauses in CNF of* $\{KB \wedge \neg\alpha\}$
new ← $\{\}$

loop do
  for each *pair of clauses* C_i, C_j ***in*** *clauses* ***do***
    resolvents ← *PL-RESOLVE*(C_i, C_j)
    if *resolvents contains* $\{\}$ ***then return*** *true*
    new ← *new* ∪ *resolvents*
  if *new clauses* ***then return*** *false*
  clauses ← *clauses* ∪ *new*

- Given formula in conjunctive normal form, repeat:
- Find two clauses with complementary literals,
- Apply resolution,
- Add resulting clause (if not already there)
- If the empty clause results, formula is not satisfiable
  - Must have been obtained from P and NOT(P)
- Otherwise, if we get stuck (and we will eventually), the formula is guaranteed to be satisfiable (proof in a couple of slides)

Adapted from Rusell, S., & Norvig, P. (2016)

5.2 Conjunctive Normal Form (CNF)

A sentence is in conjunctive normal form (CNF) if it is the conjunction of one or more clauses, where a clause is a disjunction of literals. A set of sentences is in CNF if each sentence is in CNF.

- $A \wedge B$
- $\neg A \wedge \neg B$
- $(A \vee B) \wedge (\neg C)$

Adapted from Castillo, E. et al. (2012); Rusell, S., & Norvig, P. (2016)

5.2 Conjunctive Normal Form (CNF)

A sentence is in conjunctive normal form (CNF) if it is the conjunction of one or more clauses, where a clause is a disjunction of literals. A set of sentences is in CNF if each sentence is in CNF.

Converting to CNF

1. Replace every occurrence of $\alpha \Leftrightarrow \beta$ by $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$
2. Replace every occurrence of $\alpha \Rightarrow \beta$ by $\neg\alpha \vee \beta$
3. Replace every occurrence of $\neg(\alpha \vee \beta)$ by $\neg\alpha \wedge \neg\beta$; every occurrence of $\neg(\alpha \wedge \beta)$ by $\neg\alpha \vee \neg\beta$; and every occurrence of $\neg\neg\alpha$ by α. Repeat as long as possible
4. Replace every occurrence of (α ∧ β) ∨ γ by (α ∨ γ) ∧ (β ∨ γ), and every occurrence of α ∨ (β ∧ γ) by (α ∨ β) ∧ (α ∨ γ)

Given:

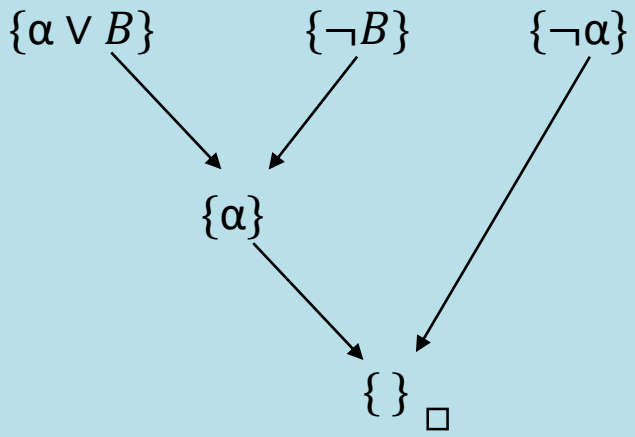
1. $\neg(p \rightarrow q) \vee (r \rightarrow p)$
2. $\neg(\neg p \vee q) \vee (\neg r \vee p)$
3. $(p \wedge \neg q) \vee (\neg r \vee p)$
4. $(p \vee \neg r \vee p) \wedge (\neg q \vee \neg r \vee p)$

Adapted from Castillo, E. et al. (2012); Rusell, S., & Norvig, P. (2016)

5.2 Resolution and Completeness

- Keep applying resolution to clauses that contain complementary literals and adding resulting clauses to the list
- Break If there are no new clauses to be added, then KB does not entail α
- Or If two clauses resolve to form an empty clause, we have a contradiction and KB ⊨ α

$\{KB \wedge \neg\alpha\}$



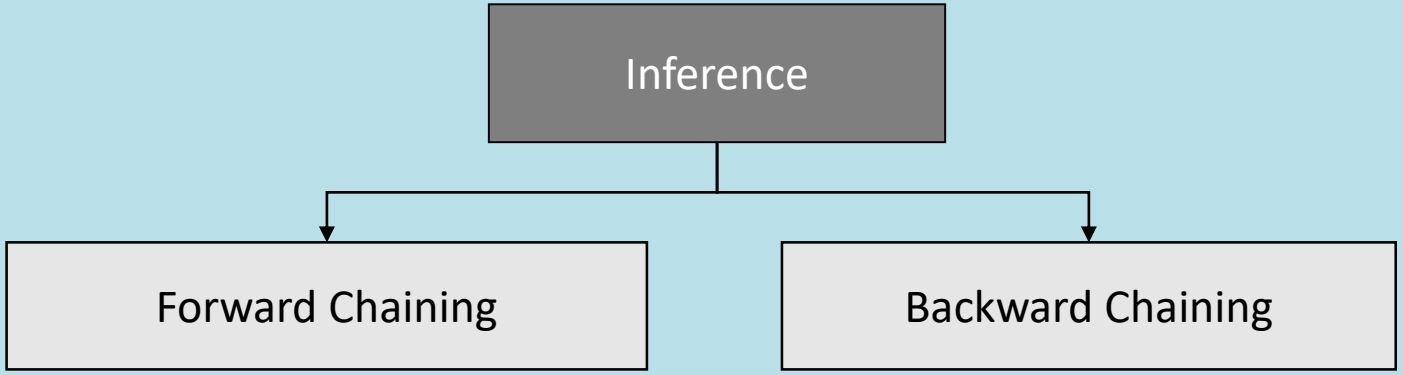
Adapted from Castillo, E. et al. (2012); Rusell, S., & Norvig, P. (2016)

5.2 Inference in Logical Systems

- So far, we know the modus ponens for simple 1-rule inference

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta} \quad \Rightarrow \quad \frac{\textbf{if } MANUAL-CHECK = FAILED, \textbf{ then } ACCEPT = NO \qquad MANUAL-CHECK = FAILED}{ACCEPT = NO}$$

- However, rule-based Systems can use multiple rules for inference. There exists three popular approaches for that:



5.2 Horn Clauses and Definite Clauses

- Horn clauses are implications with only positive literals

$$x_1 \, AND \, x_2 \, AND \, x_4 \; => \; x_3 \, AND \, x_6$$

$$TRUE \; => \; x_1$$

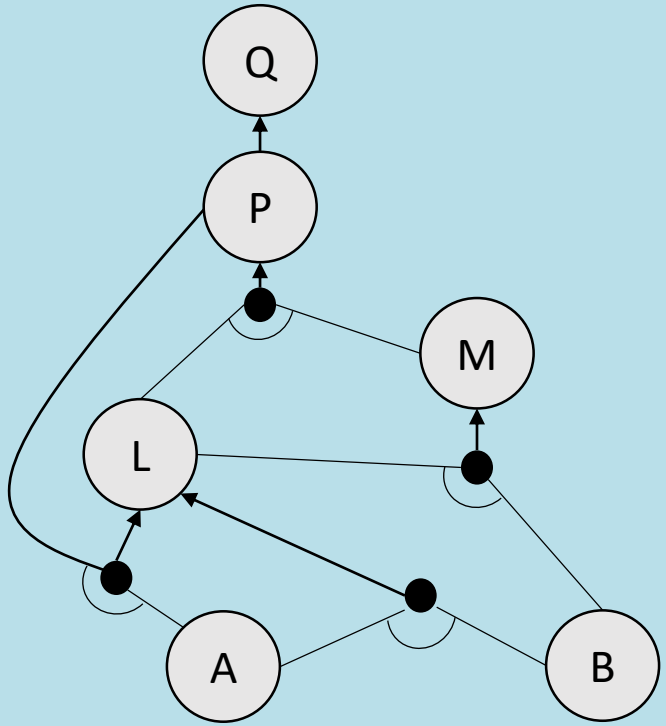
- A definite clause is a disjunction with exactly one positive literal

- Knowledge bases with only definitive clauses have many positive characteristics, which allow efficient inference mechanisms (see Russel & Norvig, 2016)

Adapted from Castillo, E. et al. (2012); Rusell, S., & Norvig, P. (2016)

5.2 Forward Chaining

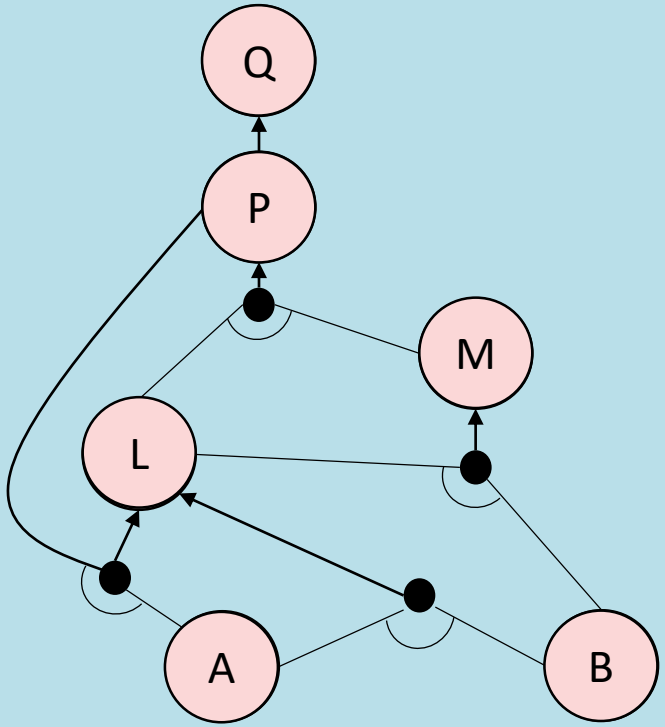
- **Idea**: find any rule whose premises are satisfied in the *KB*, add its conclusion to the *KB*, and keep going until query is found



$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow \mathrm{P}$$
$$B \wedge L \Rightarrow \mathrm{M}$$
$$A \wedge P \Rightarrow \mathrm{L}$$
$$A \wedge B \Rightarrow \mathrm{L}$$
$$A$$
$$B$$

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014); Rusell, S., & Norvig, P. (2016)

5.2 Forward Chaining Animation

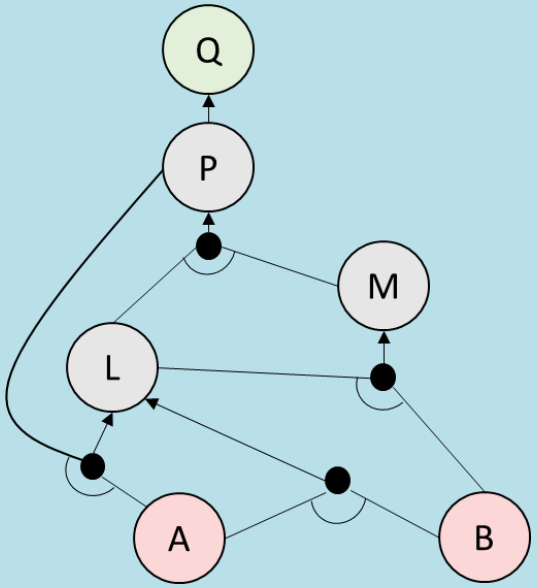


$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow \mathrm{P}$$
$$B \wedge L \Rightarrow \mathrm{M}$$
$$A \wedge P \Rightarrow \mathrm{L}$$
$$A \wedge B \Rightarrow \mathrm{L}$$
$$A$$
$$B$$

Adapted from Castillo, E. et al. (2012); Rusell, S., & Norvig, P. (2016)

5.2 Backward Chaining

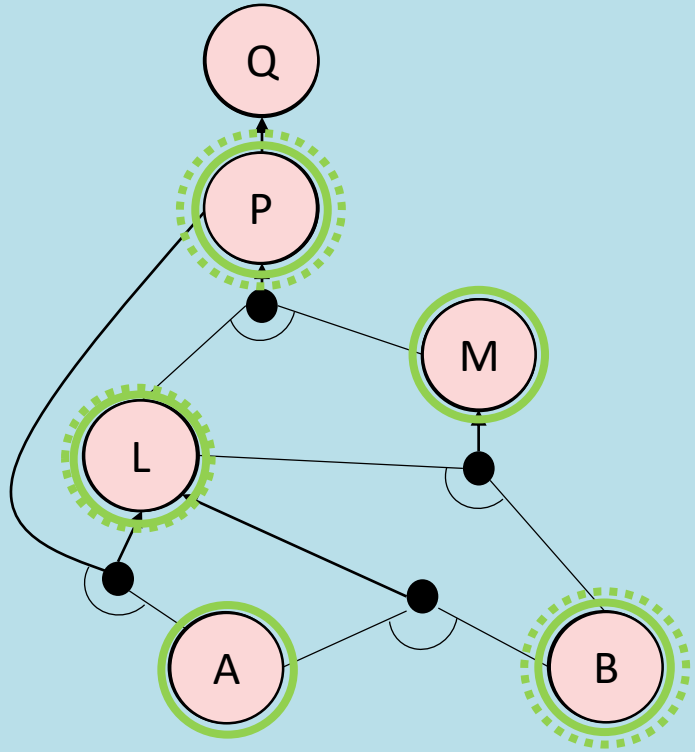
- **Idea**: work backwards from the query q to prove q,
  - check if q is known already or
  - prove all premises of some rule concluding q



$$P \Rightarrow Q$$
$$L \wedge M \Rightarrow \mathrm{P}$$
$$B \wedge L \Rightarrow \mathrm{M}$$
$$A \wedge P \Rightarrow \mathrm{L}$$
$$A \wedge B \Rightarrow \mathrm{L}$$
$$A$$
$$B$$

Adapted from Castillo, E. et al. (2012); Beierle, C., & Kern-Isberner, G. (2014); Rusell, S., & Norvig, P. (2016)

5.2 Backward Chaining Animation



$$P \Rightarrow Q$$

$$L \wedge M \Rightarrow \mathrm{P}$$

$$B \wedge L \Rightarrow \mathrm{M}$$

$$A \wedge P \Rightarrow \mathrm{L}$$

$$A \wedge B \Rightarrow \mathrm{L}$$

$$A$$

$$B$$

5.2 Forward vs. backward chaining

- Forward chaining is data-driven, automatic processing
  - May do lots of work that is irrelevant to the goal
- Backward chaining is goal-driven, appropriate for problem-solving
  - Complexity can be much less than linear in size of KB

Your turn!

Task

Your colleague Sarah plans to buy a new sportcar, but she can not decide which one. You remember your AI course and the chapter about logic and try to help her. Model the following problem with propositional logic:

- She wants to buy a Porsche but struggeles between 911 (N), Panamera (P) and Taycan (T)
- She says that she likes shopping (S), and hence she doesn't want buy a two-door car like the 911 because it has not enough place for shopping
- If you want to buy a Taycan, you need an electric fueling station in your garage (E)

She now assumes (H_1):

- If you like shopping, and have no electric fueling station you have to buy a Panamera

Modell this problem with propositional logic. Can you "prove" her assumption H_1? Explain.

As Long as There are Users of outdated Techniques, There is a Need of Specialist

WELT Abonnement Ticker Suche Anmelden

HOME » WIRTSCHAFT » WEBWELT & TECHNIK » Cobol: USA suchen wegen Coronakrise Programmierer für alte Systeme

WIRTSCHAFT

SMART LIVING STELLENMARKT KARRIERE DIGITAL GELD MITTELSTAND

WEBWELT & TECHNIK COBOL IN DER CORONAKRISE

USA suchen dringend Programmierer für uralte Behördensysteme

Veröffentlicht am 27.04.2020 | Lesedauer: 5 Minuten

Von Kathrin Stoll

79

Welt Online (2020)

Exercises

Workbook Exercises

- Please read the chapters 7-12 from Rusell, S., & Norvig, P. (2016) to understand the concepts and algorithms of knowledge reasoning and representation Then work through the related exercises. Focus on the contents we discussed in lecture.
- Read and understand the Wumpus World Problem from chapter 7. Try to model it with the concepts from this lecture.

Coding Exercises

- Try to build a simple rule-based agent with Python by implementing the decision rules a simple if-else checks. What is the problem of this approach if you think at expanding the knowledge based? Make notes and prepare for the lectorial of this chapter, we will find a more convinient way to process and represent knowledge in Python

References

Literature

1. Beierle, C., & Kern-Isberner, G. (2014). Methoden wissensbasierter Systeme: Grundlagen, Algorithmen, Anwendungen. Springer-Verlag.
2. Brachman, R. J., Levesque, H. J., & Reiter, R. (Eds.) (1992): *Knowledge representation*. MIT press.
3. Castillo, E., Gutierrez, J. M., & Hadi, A. S. (2012). Expert systems and probabilistic network models. Springer Science & Business Media.
4. Peirce, C. S. (1931): Collected papers of charles sanders peirce. Harvard University Press, 1931.
5. Rusell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach.* Global Edition.

Images

All images that were not marked other ways are made by myself, or licensed ↗CC0 from ↗Pixabay.

References

Further reading

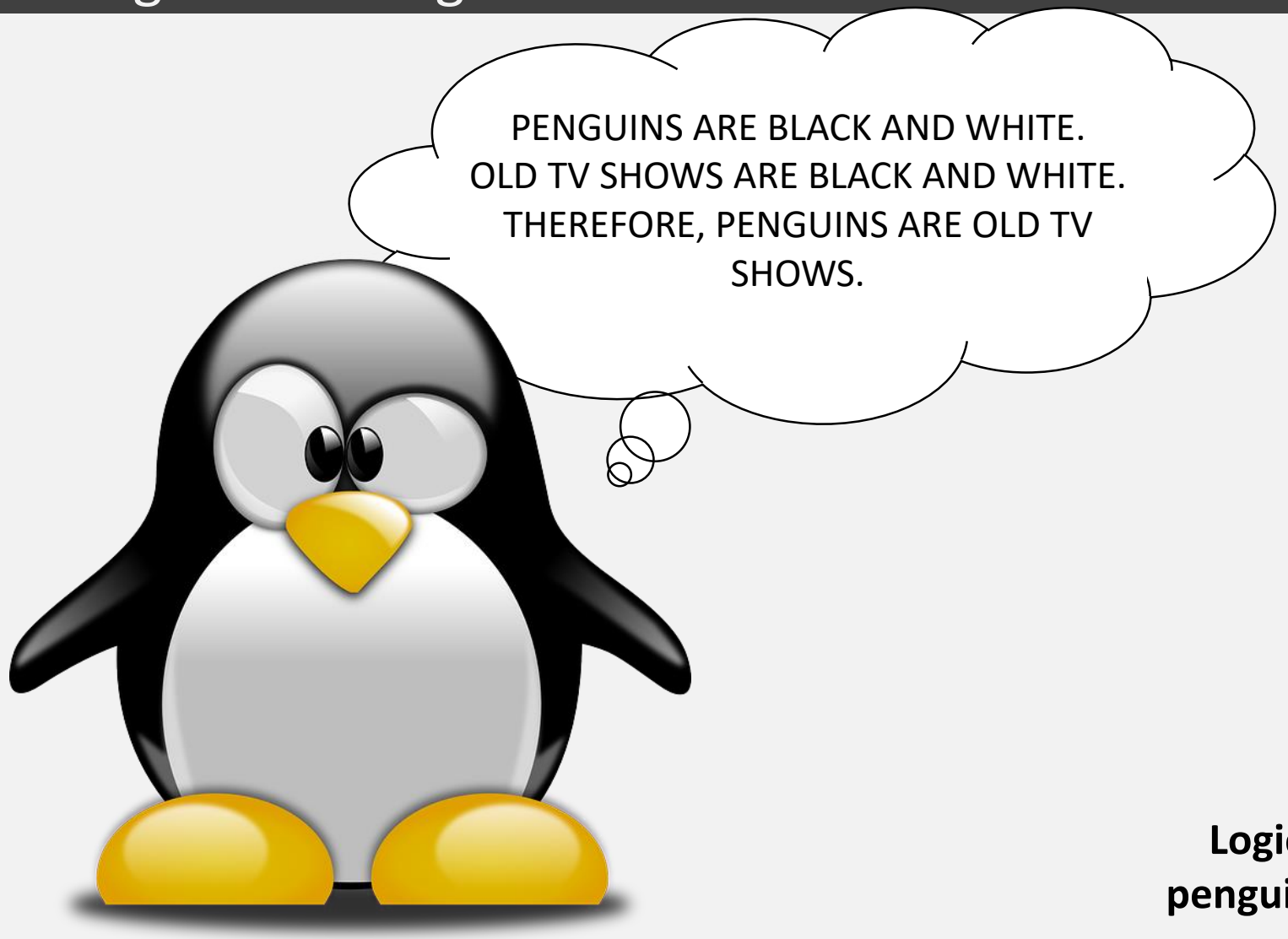
- Even if knowledge-based systems are a bit outdated, there exists still many frameworks to developp such systems with state of the art technology like Python, Java etc. If you are interested I recommend to take a look at the following frameworks and libraries:
  - C Language Integrated Production System (CLIPS): An environment used to make rule or object based expert systems developed by the NASA (↗www.clipsrules.net)
  - Python Knowledge Engine Pyke (Pyke), which allows you to use logic programming to make expert systems in Python (↗www.pyke.sourceforge.net)
  - D3web, Java Knowledge Base System that uses XML (↗www.d3web.de)

- The popular game Age of Empires II from Microsoft Game Studios relies on CLIPS for its computer AI. I can recommend you to take a look at the guide from redmechanic (on ↗ Steam), from Leif Ericson(↗ aok.heavengames.com), or a copy of Microsoft‘s „Computer Player Strategy Builder Guide” from my git hub repository (Media/CPSB.zip) to implement your own knowledge-based AI. Play against it and try to win against yourself ;-)

Glossary

Entailment	*Necessary truth of one sentence given another*
Inference	*Deriving sentences from other sentences*
Logical Agents	*Logical agents apply inferenceto a knowledge baseto derive new information and make decisions*
Knowledge Base	*The component of an expert system that constains the systems‘s knowledge organized in collection of facts about the system‘s domain.*
Knowledge Representation	*The study of how “what we know” can at the same time be represented as comprehensibly as possible and reasoned with as effectively as possibly from an information system*
Soundness	*Derivations produce only entailed sentences*
Completeness	*Derivations can produce all entailed sentences*

When Knowledge-Based Penguins Fail…



Logic: another thing that penguins aren‘t very good at.

Adapted from Glasbergen | Image source: ↗ Pixabay (2019) / ↗ CC0