CHAPTER

Nonlogical Methods for Commonsense Reasoning 17

This book uses the event calculus for commonsense reasoning, which is based on predicate logic. Why logic? What about other approaches to commonsense reasoning? In this chapter, we review several other important approaches to commonsense reasoning: qualitative reasoning, analogical processing, probabilistic reasoning, and the society of mind. We assess their strengths and weaknesses and discuss their relationships to the event calculus.

17.1 QUALITATIVE REASONING

Qualitative reasoning is concerned with explaining and predicting the behavior of physical systems. The task is to take as input a description of a physical system and an initial state and produce as output the possible behaviors of the system given the initial state. Some examples of physical systems modeled using this approach include a water pressure regulator, two tanks connected by a pipe, heat flow between two objects, boiling water in a pot, and a block connected to a spring. Several schemes for qualitative reasoning have been developed. In this section, we describe qualitative simulation (QSIM) which is representative.

17.1.1 QSIM

In QSIM, a physical system is described by a qualitative differential equation (QDE), which corresponds to an infinite set of ordinary differential equations. A QDE consists of a set of variables and a set of constraints on those variables. Possible constraints are $x + y = z, xy = z, x = -y, y = \frac{dx}{dt}, M^+(x, y)$, and $M^-(x, y)$, where x, y, and z are variables. The constraint $M^+(x, y)$ means that x is monotonically increasing with y, and $M^-(x, y)$ means that x is monotonically decreasing with y. Each variable is associated with landmark values $l_1 < \cdots < l_n$. A *qualitative value* of a variable consists of a landmark value l_i or interval (l_i, l_{i+1}) and a direction, which is increasing, steady, or decreasing. A state assigns a qualitative value to each variable.

Given a QDE and an initial state, the QSIM algorithm produces a *behavior tree*, or tree of states. The QSIM algorithm operates by repeatedly taking a state and generating successor states that satisfy the constraints. QSIM also allows a physical system to be described by multiple QDEs and transitions between the QDEs. For example, a bouncing ball can be modeled using one QDE for motion through the air and another QDE for bouncing.

Qualitative reasoning can be used for commonsense reasoning about physical systems. But general commonsense reasoning requires reasoning about both continuous and discrete change. Qualitative reasoning is largely concerned with continuous change and is not concerned with discrete events and associated commonsense phenomena such as context-sensitive, indirect, and nondeterministic effects of events. The commonsense law of inertia is not represented (or even needed) in qualitative reasoning because the behaviors of the system are determined by the QDE.

It has been shown how qualitative descriptions of systems can be expressed in first-order logic and how system behaviors can be predicted using deduction. Some work has been performed on incorporating QSIM into the event calculus; see Exercises 17.2 and 17.3.

17.2 ANALOGICAL PROCESSING

A person who encounters a novel situation may not already have the commonsense knowledge necessary to reason about that situation. In such a case, the person might be able to reason about the novel situation by analogy to a familiar situation. For example, consider the case of sand being poured into a kitchen sink with the stopper in place. By analogy to the case of water, we can predict that the sand level will increase and that the sand will eventually spill onto the floor.

Some researchers have proposed that analogical processing can be used for commonsense reasoning and, moreover, that commonsense reasoning often involves analogical processing. In this section, we examine a particularly well-developed mechanism for analogical processing, the structure-mapping engine.

17.2.1 STRUCTURE-MAPPING ENGINE

The structure-mapping engine (SME) takes as input a base domain and a target domain and produces as output zero or more global mappings from the base domain to the target domain. A domain consists of a set of labeled ordered trees, or rooted trees such that (1) each node has an associated label and (2) the children of each node are ordered. Leaf nodes represent objects such as *Beaker* and *Coffee*. Nonleaf nodes represent functions such as *Pressure* and *Temperature*, attributes such as *Liquid* and *Red*, and relations such as *Causes*, *Implies*, and *GreaterThan*. The children of a nonleaf node represent the arguments of the function, attribute, or relation. A function node has one or more children, an attribute node has one child, and a relation node has two or more children.

Global mappings

A global mapping consists of (1) a one-to-one mapping M from nodes of the base domain to nodes of the target domain satisfying certain constraints, (2) candidate inferences in the target domain, and (3) a score. The mapping M satisfies the following constraints:

- M maps object nodes to object nodes, function nodes to function nodes, attribute nodes to attribute nodes, and relation nodes to relation nodes.
- If M maps a base function or attribute node b to a target node, then b is not a root node. That is, function and attribute nodes are not mapped unless they are part of a larger structure.
- If M maps a base attribute or relation node b to a target node t, then b and t have the same label. Note that this constraint does not apply to object and function nodes.
- If M maps a function, attribute, or relation node b to a target node t, then M maps each child of b to a child of t.

Generating candidate inferences

In order to generate candidate inferences in the target domain, we start by forming the set C_b of all trees B of the base domain such that M maps any nonleaf node of B to a target node. That is, we form the set C_b of base domain trees having one or more mapped function, attribute, or relation nodes. These are the base domain trees involved in the analogy. Then, the target domain candidate inferences C_t consist of the base domain trees C_b with their object and function nodes relabeled using the terminology of the target domain. Specifically, for each object and function node b of C_b, we do the following: If M maps b to a target node t, then we relabel b with the label of t; otherwise, if M does not map b to any target node, then we relabel b with a new unique label.

The *score* is an estimate of the quality of the mapping M. A higher score is assigned to the extent that larger trees are mapped, base nodes are mapped to target nodes with the same label, and children of a base node are mapped to children of a target node in the right order.

Example

SME handles over 40 examples, one of which involves an analogy between water flow and heat flow. The base domain, water flow, is represented as follows:

$$Causes(GreaterThan(Pressure(Beaker), Pressure(Vial))$$
$$Flow(Beaker, Vial, Water, Pipe))$$
$$GreaterThan(Diameter(Beaker), Diameter(Vial))$$
$$Liquid(Water)$$
$$FlatTop(Water)$$

The target domain, heat flow, is represented as follows:

$$GreaterThan(Temperature(Coffee), Temperature(IceCube))$$
$$Flow(Coffee, IceCube, Heat, Bar)$$
$$Liquid(Coffee)$$
$$FlatTop(Coffee)$$

Given these base and target domains, SME produces three global mappings. The global mapping with the highest score (=5.99) includes the following mapping M:

$$\begin{gathered}
Beaker \mapsto Coffee \\
Vial \mapsto IceCube \\
Water \mapsto Heat \\
Pipe \mapsto Bar \\
Pressure(Beaker) \mapsto Temperature(Coffee) \\
Pressure(Vial) \mapsto Temperature(IceCube) \\
GreaterThan(Pressure(Beaker), Pressure(Vial)) \mapsto \\
GreaterThan(Temperature(Coffee), Temperature(IceCube)) \\
Flow(Beaker, Vial, Water, Pipe) \mapsto \\
Flow(Coffee, IceCube, Heat, Bar)
\end{gathered}$$

This global mapping includes the following candidate inference:

$$\begin{gathered}
Causes(GreaterThan(Temperature(Coffee), Temperature(IceCube)), \\
Flow(Coffee, IceCube, Heat, Bar))
\end{gathered}$$

Discussion

SME is not intended to be a complete method for commonsense reasoning. Although SME can find potential analogies and generate candidate inferences, it does not address the crucial problem of how to evaluate and possibly repair the inferences. It is assumed that other commonsense knowledge and commonsense reasoning mechanisms will be used to perform these tasks. An interesting line of research would be to integrate SME and the event calculus; see Exercise 17.4.

17.3 PROBABILISTIC REASONING

Some researchers have proposed the use of probability theory for commonsense reasoning. By using probability measures or other measures of uncertainty, we can quantify not only our uncertainty about a particular scenario but also our uncertainty about our knowledge of how the world works. We can quantify the degree to which we are certain about our formalization of a domain, such as how sure we are that a given action has given preconditions or effects. Using probabilistic reasoning, we can quantify our uncertainty about various sorts of commonsense conclusions derived from information that is given.

17.3.1 PROBABILITY AND ACTION

Let us consider how probability theory can be used to reason about a simple action, turning on a faucet. In order to reason about properties over time, we introduce a random variable for each relevant property and timepoint or time interval. We represent the fact that the faucet is running at time 0 as a Boolean random variable R_0 and the fact that the faucet is running at time 1 as a Boolean random variable R_1.

In order to reason about events that occur in time, we introduce a random variable for each relevant event and timepoint or time interval. We represent the event of turning on the faucet at time 0 as a Boolean random variable T_0.

Then we represent our knowledge of this domain. We do not specify a full joint distribution. Instead, we assume that for every time t and random variable v at time t:

- v is directly influenced by zero or more random variables V at time $t-1$.
- the random variable v and the random variables at all times less than or equal to $t-2$ are conditionally independent given the random variables V.

In order to represent the effects of events, we specify the conditional probabilities of properties at time t given the properties and events at time $t-1$. We consider the influence of R_0 and T_0 on R_1. We specify that, if a faucet that is not running is turned on, it will be running with high probability:

$$P(R_1|\neg R_0, T_0) = 0.98 \tag{17.1}$$

(Here we deal only with the times 0 and 1. In a formulation with n times, we would write $P(R_t|\neg R_{t-1}, T_{t-1}) = 0.98$ for each $t \in \{1, 2, \ldots, n-1\}$.) If a faucet that is already running is turned on, it will be running with high probability:

$$P(R_1|R_0, T_0) = 0.99$$

Furthermore, we represent what happens when events do not occur. That is, we represent instances of the commonsense law of inertia. A faucet that is not running and is not turned on will not suddenly start running:

$$P(R_1|\neg R_0, \neg T_0) = 0.01 \tag{17.2}$$

A faucet that is running and is not turned on will continue to run:

$$P(R_1|R_0, \neg T_0) = 0.95$$

We can then perform temporal projection, abduction, and postdiction. For temporal projection, we may ask whether the faucet is running at time 1 given that it is not running at time 0 and is turned on at time 0. We see immediately from (17.1) that this is very likely.

For abduction, we may ask whether the faucet is turned on at time 0 given that it is not running at time 0 and is running at time 1. We must determine

$$P(T_0|\neg R_0, R_1)$$

From the definition of conditional probability, we have

$$P(T_0|\neg R_0, R_1) = \frac{P(R_1, \neg R_0, T_0)}{P(\neg R_0, R_1)} \tag{17.3}$$

Using the chain rule, we have

$$P(R_1, \neg R_0, T_0) = P(R_1|\neg R_0, T_0)P(\neg R_0|T_0)P(T_0)$$

From this and (17.1), we have

$$P(R_1, \neg R_0, T_0) = 0.98 \cdot P(\neg R_0|T_0)P(T_0) \tag{17.4}$$

At this point we cannot proceed without further information. We first make an assumption regarding the independence of R_0 and T_0:

$$P(R_0|T_0) = P(R_0) \tag{17.5}$$

(In a more detailed model, we might not wish to make such an assumption. For example, an agent may be unlikely to turn on a faucet that is already running.) We must also specify prior probabilities for the faucet running at time 0 and the faucet being turned on at time 0:

$$P(R_0) = 0.5 \tag{17.6}$$

$$P(T_0) = 0.1 \tag{17.7}$$

From (17.4), (17.5), (17.6), and (17.7), we have

$$P(R_1, \neg R_0, T_0) = 0.98 \cdot 0.5 \cdot 0.1 = 0.049 \tag{17.8}$$

From the product rule, we have

$$P(\neg R_0, R_1) = P(R_1|\neg R_0)P(\neg R_0) \tag{17.9}$$

By expanding the possible cases for T_0, we have

$$P(R_1|\neg R_0) = P(R_1|\neg R_0, \neg T_0)P(\neg T_0|\neg R_0) + P(R_1|\neg R_0, T_0)P(T_0|\neg R_0)$$

From this, (17.2), (17.5), (17.7), and (17.1), we have

$$P(R_1|\neg R_0) = 0.01 \cdot 0.9 + 0.98 \cdot 0.1 = 0.107$$

From this, (17.9) and (17.6), we have

$$P(\neg R_0, R_1) = 0.107 \cdot 0.5 = 0.0535$$

From this, (17.3), and (17.8), we have

$$P(T_0|\neg R_0, R_1) = \frac{0.049}{0.0535} = 0.9159$$

Thus, it is likely that the faucet is turned on at time 0.

For postdiction, we may calculate the probability that the faucet is not running at time 0 given that it is turned on at time 0 and is running at time 1:

$$P(\neg R_0|T_0, R_1) = 0.4975$$

This calculation is performed in a fashion similar to the calculation for abduction.

This is a simple example involving just three random variables. It is apparent that using these techniques to perform commonsense reasoning on larger examples will be cumbersome.

17.3.2 BAYESIAN NETWORKS

A more convenient way of performing these calculations is to use Bayesian networks. A *Bayesian network* is a directed acyclic graph in which each node represents a random variable, and a directed link from node n_1 to node n_2 represents that n_1 directly influences n_2. Each node n is associated with a conditional probability table that specifies the conditional probability for every value of n given every combination of values of every node that directly influences n or the prior probability of n if no nodes directly influence n.

Still, even using Bayesian networks, we must introduce a node for every property and event at every timepoint or time interval and specify the conditional probability table for every node. There are many probabilities to specify. In our faucet example, we had to specify probabilities for each instance of the commonsense law of inertia, because probability theory does not supply us with any general mechanism for representing this law.

17.4 SOCIETY OF MIND

Marvin Minsky developed the *society of mind* theory of human cognition, which proposes that the flexibility of human intelligence arises from the use of many different techniques, not simply one technique such as analogy, connectionism, logic, or probability. This theory starts with the following notions.

The mind consists of a vast collection of interacting simple processes, called agents, critics, or resources.

The agents use many different types of representations and many different methods of reasoning in order to exploit the advantages of each representation and reasoning method and to avoid their disadvantages.

Representations in different agents are connected to one another. These connections facilitate the simultaneous use of representations and switching between representations whenever difficulties are encountered with one of the representations.

The agents are concerned with various realms; examples include the aesthetic, physical, possessional, psychological, spatial, temporal, and topological realms. One or more agents may be concerned with any given realm.

Some agents are concerned with *reflection* or monitoring and influencing other agents. For example, reflective agents can notice other agents getting into loops and help them break out of those loops.

Minsky proposes various constructs that can be used to build up a society of mind.

A *frame* consists of slots and default values for the slots used to capture expectations about a frequently encountered class of situations. When a frame is applied to a situation, the default values may be used as is or they may be modified to match the situation.

A *frame-array* is a collection of frames that share one or more slots. The frames represent a class of situations from different perspectives, or they represent related classes of situations. The shared slots enable switching from one frame to another to facilitate shifting perspectives or situations.

A *transframe* is a type of frame that has slots for an origin, action, and destination and is used to represent the effects of actions. Transframes may be chained together by matching the destination of each transframe with the origin of the next transframe to predict the results of a sequence of actions.

A *K-line* is a memory structure that records what agents were used to solve a problem. The K-line is used to activate those agents when solving similar problems in the future.

A *paranome* is a mapping among alternative representations of a situation used by two or more agents. When one of the agents updates its representation, the paranome allows the other agents to update their representations appropriately.

Other constructs proposed by Minsky include censors, level-bands, micronemes, picture-frames, polynemes, pronomes, proto-specialists, and suppressors.

There are two ways of viewing the relationship between the society of mind and the event calculus:

- The event calculus could be viewed as one effective method for performing commonsense reasoning within a larger society of mind.
- Constructs from the society of mind could themselves be integrated into the event calculus—see Exercise 17.6.

In this section, we review three commonsense reasoning systems inspired by the society of mind: ThoughtTreasure, Polyscheme, and EM-ONE. All three systems involve collections of interacting agents. ThoughtTreasure and Polyscheme use multiple representation and reasoning methods. Polyscheme and EM-ONE use reflection.

17.4.1 THOUGHTTREASURE

The ThoughtTreasure commonsense reasoning and story understanding architecture/system uses multiple representation and reasoning mechanisms, including finite automata, grids, path planning, logical formulas, theorem proving, and scripts. Reasoning is performed by a collection of agents that communicate over a shared data structure.

Commonsense domains

ThoughtTreasure represents and reasons about objects, time-varying properties, events, time, space, and goals. Time is represented using timepoints and time intervals. If t_1 and t_2 are timepoints, e is an event, and p is a property, then

$$@t_1 : t_2 \,|\, e$$

represents that e occurs from t_1 to t_2, and

$$@t_1 : t_2 \,|\, p$$

represents that p is true from t_1 to t_2. Space is represented by a network of *grids* (occupancy arrays). A physical object occupies one or more cells of a grid over a given time interval.

Reasoning types

ThoughtTreasure performs two types of reasoning: simulation and understanding. In simulation, the system takes as input a state of the world in which a character has an active goal and produces as output the events and time-varying properties that follow from that state of the world. The system handles several benchmark problems involving simulation: moving a penny from one jar to another, traveling from Paris to New York, and watching television. In the television scenario, the system is told that a character Jim is located in a particular apartment grid containing various objects such as appliances and furniture. The system then produces a sequence of events in which Jim walks from his initial location in the grid to the television and turns it on.

In understanding, the system takes natural language text as input and parses the text into properties and events. Given these observations, the system produces as output zero or more models that tally with those observations. Each model consists of the input properties and events augmented with properties and events filled in by the system. Several benchmark problems involving understanding are handled: going to sleep, stepping outside a grocery store, and taking a shower. In the shower scenario, the system is told that Jim woke up and poured shampoo on his hair. The system produces a model in which Jim gets out of bed, walks to the shower, turns it on, picks up the shampoo, and pours it on his hair.

Discussion

ThoughtTreasure has no general mechanism to deal with the commonsense law of inertia. The burden falls on each agent to update representations appropriately. For example, if Jim is known to be at one location starting at time 0:

```
@0:na|[at-grid Jim AptGrid <gridsubspace 20 78>]
```

and the `grid-walk` agent simulates Jim walking to another location from time 2 to time 3:

```
@2:3|[grid-walk AptGrid 20 78 16 82]
```

then that agent updates the representations to reflect that Jim was at the first location from time 0 to time 3 and at the second location starting at time 3:

```
@0:3|[at-grid Jim AptGrid <gridsubspace 20 78>]
@3:inf|[at-grid Jim AptGrid <gridsubspace 16 82>]
```

Thus, knowledge of the effects of events is represented procedurally within agents. Similarly, knowledge about context-sensitive effects and preconditions of actions are represented procedurally within agents. For example, the agent that simulates a television represents that the television will not go on unless it is plugged in.

ThoughtTreasure does not have a general mechanism to deal with the indirect effects of events. Just as for the commonsense law of inertia, individual agents are responsible for performing the appropriate updates. For example, when the `grid-walk` agent simulates a character walking from one grid location to another, the agent moves not only the character but also all objects held by the character and all objects inside those objects.

ThoughtTreasure does not address continuous change, nondeterministic effects, or release from the commonsense law of inertia. Although ThoughtTreasure is able to represent concurrent events, it does not address concurrent events with cumulative and canceling effects.

ThoughtTreasure is effective at simulation, but less effective at understanding. Both simulation and understanding are performed by agents. Although it is relatively easy to write agents for simulation, it is more difficult to write agents for understanding. Agents must be written to coordinate with one another in order to agree on an interpretation of the input.

ThoughtTreasure does not have a general mechanism for dealing with the problem of agent coordination. This problem is taken up in Polyscheme.

Agent coordination is not an issue in the event calculus because it does not use agents. Instead of representing knowledge procedurally within agents, the approach in the event calculus is to represent knowledge declaratively as logical formulas and to use off-the-shelf procedures for reasoning. The declarative approach allows us to use the best available search techniques, as integrated into off-the-shelf solvers. A procedural knowledge representation requires the procedures to deal with knowledge representation as well as search—a tall order.

17.4.2 POLYSCHEME

The Polyscheme cognitive architecture/system uses multiple representation and reasoning techniques, including scripts, transframes, hash tables, constraint graph propagation, rules, and single-layer neural networks. Reasoning is performed by a collection of communicating specialists: perception, attribute, physics, causal, temporal, identity hypothesis, identity, place, path, and reflective specialists. Each specialist uses its own techniques for representing and reasoning about the effects of events, context-sensitive effects of events, event preconditions, triggered events, and other commonsense phenomena.

Commonsense domains

Polyscheme represents and reasons about objects, object identity, time-varying properties, events, time, and space. Although specialists may use their own representations internally, specialists communicate using logical propositions. A proposition consists of a predicate, arguments, time interval, and world. Worlds, which are similar to the situations of the situation calculus, are used for hypothetical reasoning. Space is represented as a three-dimensional grid. Thus, for example, the proposition

```
at(x, p1-1-1, t1, R)
```

says that object `x` is at grid location `p1-1-1` over time interval `t1` in world `R`.

Reasoning types

The Polyscheme system performs both temporal projection and explanation. It handles the following temporal projection benchmark problems involving gravity:

> *A block starts to fall in back of a screen. Infer that the block must have fallen and hit the floor.*
>
> *If you are then told that another block was in back of the screen, infer that the first block must have fallen and hit the second block.*

Polyscheme handles the following explanation benchmark problems involving object identity:

> *A dry mouse walks behind a screen where there is known to be a puddle of water, and then a dry mouse emerges. Infer that the first and second mouse are not the same mouse.*
>
> *A red ball rolls behind a screen, and then a red ball rolls out. Infer that the first and second ball are the same ball.*
>
> *A ball rolls behind one screen, and then a ball rolls out of another screen that is separated from the first screen. Infer that the first and second ball are not the same ball.*
>
> *A wood ball and a salt ball roll behind a screen. Then a ball rolls out, moves into a puddle of water, and does not melt. Infer that the ball that rolled out was the wooden ball.*

(We show how the event calculus can be used to solve similar problems of object identity in Section 10.3.)

Focus mechanism

The coordination of the activities of specialists is accomplished in Polyscheme via a focus mechanism. At any moment, all specialists focus on a single proposition. A *focus* consists of a proposition and a focus mode, which is PERCEIVE, ASSERT, DENY, WONDER, FIND, ELABORATE, or IMAGINE. An *attraction* consists of a focus and charge. Polyscheme conducts a search for a solution to a commonsense reasoning problem using the following algorithm:

1. The specialists issue attractions.
2. The current focus is set to be the focus of the most highly charged attraction.
3. The specialists concentrate on the current focus:
   (a) Each specialist informs the reflective specialist whether it considers the truth value of the proposition to be true, false, or unknown.
   (b) The reflective specialist determines whether the specialists agree or disagree.
   (c) If the specialists disagree, then the reflective specialist issues attractions designed to move the specialists toward agreement. For example, the reflective specialist may issue an attraction to try to perceive the truth value of the proposition in the external environment or to imagine the consequences of the proposition being false.
4. Go to step 1.

By setting the charges appropriately, various algorithms such as depth-first search, breadth-first search, stochastic simulation, and truth maintenance can be implemented or approximated. In contrast to systems that switch representation techniques only when a difficulty is encountered, Polyscheme is able to make use of all representation techniques implemented by specialists at every step of the search.

Discussion

Polyscheme does not have a general way of dealing with the commonsense law of inertia. The burden is placed on the individual specialists to make the appropriate inferences. Polyscheme does not address continuous change, indirect effects, or nondeterministic effects. Polyscheme can represent concurrent events, but does not deal with concurrent events with cumulative and canceling effects.

Polyscheme improves on ThoughtTreasure by providing a focus mechanism to coordinate the activities of the specialists. An important problem in need of further research is how to set the attractions and charges in such a way as to avoid classical problems of search such as combinatorial explosion of alternatives, getting stuck at local maxima, and oscillation.

17.4.3 EM-ONE

The EM-ONE architecture/system for reflective commonsense thinking has its roots in the multilayer cognitive architectures of Marvin Minsky and Aaron Sloman. EM-ONE consists of a set of critics and a library of narratives or cases. Critics observe, diagnose, and repair problems both in the external world and in the internal mental world of the system. *Narratives* are representations of stories in a simulated world that serve as a source of commonsense knowledge for use by the critics.

Critics

A critic consists of a condition pattern, a narrative pattern, and an action. When a critic is invoked, its patterns are evaluated. If the condition pattern matches the current conditions and the narrative pattern matches a narrative, then the critic is engaged and its action is performed. Four types of critics are implemented in EM-ONE: reactive critics, deliberative critics, reflective critics, and metacritics.

Reactive critics propose and execute actions in the world in response to world conditions. For instance, one reactive critic in EM-ONE executes an immediate turn toward a person who has just called for help.

Deliberative critics reason about a situation before taking action. The critics conduct a heuristic search through a space of hypothetical worlds. Some deliberative critics in EM-ONE evaluate hypothetical worlds, and others generate new ones. Examples of critics that evaluate worlds are those that look for unachieved goals or unexpected consequences of actions. Examples of critics that generate new worlds are those that consult narratives in order to infer the effects of an event, infer a plan that a character might select in order to achieve a goal, or infer that a character might execute the next step of a plan.

Reflective critics reason about and modify the mental processes of the system. They consult *traces* of mental activity, which are records of the critics that are invoked or engaged, and the facts that critics assert or retract. One reflective critic in EM-ONE is engaged when an action fails to produce the intended effect and the examination of the trace reveals that a narrative containing a precondition for that action was not consulted. The reflective critic then modifies the critic that failed to consult the narrative so that it will take this narrative into account in the future.

Metacritics are used to manage reactive, deliberative, and reflective critics. They implement Minsky's *critic-selector model*, in which a critic (1) notices a particular type of problem and (2) initiates an appropriate way of thinking about that type of problem.

Metacritics play a key role in the top-level control loop of EM-ONE, which is as follows:

1. Observations of the world are accepted.
2. Metacritics are invoked.
3. Some metacritics become engaged, which may lead to the invocation of reactive, deliberative, and reflective critics.
4. Some invoked critics become engaged, and those critics may invoke other critics, which may become engaged, and so on.
5. Actions are executed in the world.
6. Go to step 1.

Commonsense domains

EM-ONE represents entities such as physical objects, characters, events, properties, and relations. Its narrative library represents knowledge about the commonsense domains of space, time, and the mental and social worlds. For example, one narrative says that the character named Pink attached a stick to a board, which caused the stick to be attached to the board. Another narrative says that Pink wants the stick to be visible, which implies that Pink believes that Pink wants the stick to be visible. Facts are represented in EM-ONE using a frame-based knowledge representation language.

Reasoning types

The deliberative critics of EM-ONE perform forms of temporal projection and planning. The reflective critics perform reflective reasoning. EM-ONE handles a benchmark problem that engages all types of critics. The benchmark problem involves two characters named Green and Pink in a simulated world. The world contains a table with only two legs. The problem is for Green to attach a third leg to the table. Green, as modeled by EM-ONE, attempts to attach a third leg to the table, but this fails because there is not enough space between the table and the ground. Green then asks Pink for help. Pink, also modeled by EM-ONE, mistakenly believes that Green wants to disassemble the table and starts removing a leg from the table. Green realizes that Pink has misunderstood its goal and informs Pink of its goal. Pink lifts up the table for Green, and Green successfully attaches the third leg to the table.

Discussion

EM-ONE represents and reasons about event effects and preconditions. It does not have general mechanisms for dealing with the commonsense law of inertia or indirect effects. EM-ONE performs a novel type of reasoning, reflection, which is not addressed in this book.

BIBLIOGRAPHIC NOTES

Qualitative reasoning

Qualitative reasoning was introduced by de Kleer (1975). de Kleer and Brown (1984) introduce component models for qualitative reasoning, Forbus (1984) introduces qualitative process (QP) theory, and Kuipers (1986) introduces QSIM. Kuipers (1994) discusses QSIM in detail. A collection of readings on qualitative reasoning is provided by Weld and de Kleer (1990). Sandewall (1989a) shows how to reason about physical systems using differential equations and logic. E. Davis (1990, pp. 312-320) formalizes in first-order logic a component model of a scale, and E. Davis (1992) formalizes in first-order logic a QP theory of heating a can of water.

Analogical processing

Gentner (1983) introduces the structure-mapping theory of analogy. Falkenhainer, Forbus, and Gentner (1989) describe the structure-mapping engine (SME). Holyoak and Thagard (1989) describe another algorithm, the analogical constraint mapping engine, that incorporates constraints for semantic similarity and pragmatic centrality, in addition to structural constraints similar to those used by SME. Gentner, Holyoak, and Kokinov (2001) discuss structure-mapping and a number of other proposals for analogical thinking. We give a condensed formalization of SME that does not capture all the details of the implementation. For example, SME allows different mapping constraints to be defined by the user. The example analogy between water flow and heat flow is from Falkenhainer, Forbus, and Gentner (1989). Labeled ordered trees are discussed by Aho, Hopcroft, and Ullman (1983).

Case-based reasoning

Case-based reasoning (CBR) (Kolodner, 1993) is similar to analogical processing. In CBR, a problem (target) is solved using the following steps: (1) a *case*, or past experience (base), similar to the current problem is retrieved; (2) the case is used to propose a solution; (3) the solution is evaluated; and (4) problems with the solution are repaired. Because CBR relies on a single case or a small number of cases (Kolodner, 1993, p. 14), CBR may have difficulty with problems whose solution requires the combination of many cases. The evaluation and repair steps are difficult challenges for CBR systems (Leake, 1996, pp. 11-12, 22-27), which is similar to the situation in analogical processing. Evaluation and repair are usually performed through rule-based reasoning, although some have proposed that CBR itself can be used for evaluation and repair (Kolodner, 1993, pp. 7, 461, and 462; Leake, 1995; Sycara, 1988). It is recognized that CBR is best used in advisory systems where human users perform evaluation and repair (Leake, 1996, pp. 11, 23).

What is missing from both analogical processing and CBR is a way of performing *reasoning from first principles*, or reasoning based on general knowledge, as described in this book. Along these lines, Forbus and Gentner (1997) and Forbus (2001, pp. 36-40) have proposed that human commonsense reasoning can be explained as a tight combination of analogical processing and first-principles qualitative reasoning. They propose that reasoning from first principles is gradually learned through generalization of many situations understood through analogical processing. They suggest that reasoning from first principles extends, rather than replaces, analogical processing.

The combination of analogical processing and qualitative reasoning has been implemented in the PHINEAS program (Falkenhainer, 1988). This program takes observations of a new domain as input and learns about the domain using SME to find an analogy to a previously known domain, using the analogy to generate hypotheses about the new domain, using qualitative reasoning to generate predictions, comparing those predictions against the observations, and revising its hypotheses if necessary.

Probabilistic reasoning

Books on probabilistic reasoning and reasoning about uncertainty are by Pearl (1988b) and J. Y. Halpern (2003). Russell and Norvig (2009) review probability theory, reasoning about uncertainty, probabilistic reasoning, and probabilistic reasoning over time. Hanks and Madigan (2005) review methods for probabilistic temporal reasoning. Our treatment of turning on a faucet is based on the treatments of the Yale shooting scenario of Pearl (1988a; 1988b, pp. 509-516) and Hanks and Madigan (2005).

Bayesian networks are discussed by Pearl (1988b), J. Y. Halpern (2003), and Russell and Norvig (2009). Kushmerick, Hanks, and Weld (1995) present a framework for probabilistic planning. Baral, Tran, and Tuan (2002) introduce PAL, a probabilistic action language based on action language $\mathcal{A}$. Tran and Baral (2004a) show how to represent the probabilistic causal models of Pearl (2000) in PAL.

Society of mind

Minsky developed the society of mind approach (Minsky & Papert, 1972, pp. 92-99; Minsky, 1974, 1977, 1980, 1982, 1986, 1991a, 1994, 2006), which uses communicating agents and societies of agents, diverse formalisms and representations, and reflection. Singh (2004) reviews the society of mind approach and work inspired by it. McCarthy et al. (2002), Minsky, Singh, and Sloman (2004), Singh and Minsky (2004), and Singh, Minsky, and Eslick (2004) propose society-of-mind-like architectures for human-level intelligence. ThoughtTreasure is described by Mueller (1998). Polyscheme is described by Cassimatis (2002), Cassimatis, Trafton, Bugajska, and Schultz (2004), and Cassimatis (2006). Search is discussed by Pearl (1984), Russell and Norvig (2009), and Hoos and Stützle (2005). EM-ONE is described by Singh (2005). An architecture for reflective thinking is described by Morgan (2013). Sloman

(2001) discusses the H-Cogaff three-layer architecture, and Minsky (2006) discusses the Model Six six-level architecture. Minsky (2006) discusses the critic-selector model. Other systems based on the society of mind approach include those of Riecken (1994), Singh (1998), and Hearn (2001).

Connectionism and neural networks

There are several other approaches related to commonsense reasoning. In the connectionist or neural network approach (McClelland, Rumelhart, & PDP Research Group, 1986; Rumelhart, McClelland, & PDP Research Group, 1986), a problem is represented as the activation levels of nodes in a network of nodes and directional links, and a solution is produced by the propagation of activation in the network. Representations in neural networks may be local or distributed. In a *local representation*, a concept is represented by a single node. In a *distributed representation*, a concept is represented as the pattern of activation across many nodes.

Sun (1994) presents a connectionist architecture for commonsense reasoning. The architecture consists of two connected neural networks: a network that uses local representations for rule application and a second network that uses distributed representations for similarity matching. The architecture is able to handle examples that combine rule-based and similarity-based reasoning. For example, the system is able to reason that geese may quack because ducks quack and geese and ducks are similar (pp. 23, 82-84). The architecture does not deal with events and time (Sun, 1996, p. 198).

Narayanan (1997) presents a model of representation and reasoning inspired by high-level motor control. He introduces a procedural representation called x-schemas, similar to Petri nets (Reisig, 1985), that can be used to represent and reason about motion words such as *walk*, *run*, *stumble*, and *fall*. Narayanan (1997, sec. 4.1) shows how x-schemas can be encoded as connectionist networks. He provides (sec. 3.7.1) an algorithm for temporal projection using x-schemas and suggests (sec. 3.8.2) that postdiction could be performed using the results of Portinale (1994).

EXERCISES

17.1 Write event calculus formulas to formalize the definition of a table used by the EM-ONE system (Singh, 2005): a table consists of "four sticks attached to a board" (p. 36).

17.2 (Research Problem) Continue the work of R. Miller and Shanahan (1996) on incorporating QSIM (Kuipers, 1994) into the event calculus.

17.3 (Research Problem) Jordan (2004) describes a system that tutors students in introductory physics. The system presents a problem to the student, who enters an answer and an explanation. The system uses abductive reasoning to build a proof for the student's answer and explanation. The proof exposes problems in the student's understanding, such as the *impetus misconception*, or the notion that a force must be continually applied to an object in order for it to move. Investigate how the event calculus, extended using the techniques of R. Miller and Shanahan (1996), could be used to diagnose common misconceptions of introductory physics students.

17.4 (Research Problem) Investigate ways of integrating SME (Falkenhainer, Forbus, & Gentner, 1989) and the event calculus. One idea is as follows: Given event calculus axioms describing a base domain and a target domain, SME could be used to find mappings from the base domain axioms to the target domain axioms and suggest additional target domain axioms.

17.5 (Research Problem) Develop techniques for probabilistic reasoning further with respect to various commonsense phenomena and benchmark problems.

17.6 (Research Problem) Investigate how constructs from the society of mind (Minsky, 1986), such as frame-arrays and paranomes, could be integrated into the event calculus.