Liberatore (1997). The idea of representing a belief state with propositions can be traced to Wittgenstein (1922).

Logical state estimation, of course, requires a logical representation of the effects of actions—a key problem in AI since the late 1950s. The dominant proposal has been the **situation calculus** formalism (McCarthy, 1963), which is couched within first-order logic. We discuss situation calculus, and various extensions and alternatives, in Chapters 10 and 12. The approach taken in this chapter—using temporal indices on propositional variables—is more restrictive but has the benefit of simplicity. The general approach embodied in the SATPLAN algorithm was proposed by Kautz and Selman (1992). Later generations of SATPLAN were able to take advantage of the advances in SAT solvers, described earlier, and remain among the most effective ways of solving difficult problems (Kautz, 2006).

The **frame problem** was first recognized by McCarthy and Hayes (1969). Many researchers considered the problem unsolvable within first-order logic, and it spurred a great deal of research into nonmonotonic logics. Philosophers from Dreyfus (1972) to Crockett (1994) have cited the frame problem as one symptom of the inevitable failure of the entire AI enterprise. The solution of the frame problem with successor-state axioms is due to Ray Reiter (1991). Thielscher (1999) identifies the inferential frame problem as a separate idea and provides a solution. In retrospect, one can see that Rosenschein's (1985) agents were using circuits that implemented successor-state axioms, but Rosenschein did not notice that the frame problem was thereby largely solved. Foo (2001) explains why the discrete-event control theory models typically used by engineers do not have to explicitly deal with the frame problem: because they are dealing with prediction and control, not with explanation and reasoning about counterfactual situations.

Modern propositional solvers have wide applicability in industrial applications. The application of propositional inference in the synthesis of computer hardware is now a standard technique having many large-scale deployments (Nowick *et al.*, 1993). The SATMC satisfiability checker was used to detect a previously unknown vulnerability in a Web browser user sign-on protocol (Armando *et al.*, 2008).

The wumpus world was invented by Gregory Yob (1975). Ironically, Yob developed it because he was bored with games played on a rectangular grid: the topology of his original wumpus world was a dodecahedron, and we put it back in the boring old grid. Michael Genesereth was the first to suggest that the wumpus world be used as an agent testbed.

EXERCISES

7.1 Suppose the agent has progressed to the point shown in Figure 7.4(a), page 239, having perceived nothing in [1,1], a breeze in [2,1], and a stench in [1,2], and is now concerned with the contents of [1,3], [2,2], and [3,1]. Each of these can contain a pit, and at most one can contain a wumpus. Following the example of Figure 7.5, construct the set of possible worlds. (You should find 32 of them.) Mark the worlds in which the KB is true and those in which each of the following sentences is true:

$\alpha_2 =$ "There is no pit in [2,2]."
$\alpha_3 =$ "There is a wumpus in [1,3]."

Hence show that $KB \models \alpha_2$ and $KB \models \alpha_3$.

7.2 (Adapted from Barwise and Etchemendy (1993).) Given the following, can you prove that the unicorn is mythical? How about magical? Horned?

> If the unicorn is mythical, then it is immortal, but if it is not mythical, then it is a mortal mammal. If the unicorn is either immortal or a mammal, then it is horned. The unicorn is magical if it is horned.

7.3 Consider the problem of deciding whether a propositional logic sentence is true in a given model.

a. Write a recursive algorithm PL-TRUE?(s, m) that returns $true$ if and only if the sentence s is true in the model m (where m assigns a truth value for every symbol in s). The algorithm should run in time linear in the size of the sentence. (Alternatively, use a version of this function from the online code repository.)

b. Give three examples of sentences that can be determined to be true or false in a *partial* model that does not specify a truth value for some of the symbols.

c. Show that the truth value (if any) of a sentence in a partial model cannot be determined efficiently in general.

d. Modify your PL-TRUE? algorithm so that it can sometimes judge truth from partial models, while retaining its recursive structure and linear run time. Give three examples of sentences whose truth in a partial model is *not* detected by your algorithm.

e. Investigate whether the modified algorithm makes TT-ENTAILS? more efficient.

7.4 Which of the following are correct?

a. $False \models True$.

b. $True \models False$.

c. $(A \land B) \models (A \Leftrightarrow B)$.

d. $A \Leftrightarrow B \models A \lor B$.

e. $A \Leftrightarrow B \models \neg A \lor B$.

f. $(A \land B) \Rightarrow C \models (A \Rightarrow C) \lor (B \Rightarrow C)$.

g. $(C \lor (\neg A \land \neg B)) \equiv ((A \Rightarrow C) \land (B \Rightarrow C))$.

h. $(A \lor B) \land (\neg C \lor \neg D \lor E) \models (A \lor B)$.

i. $(A \lor B) \land (\neg C \lor \neg D \lor E) \models (A \lor B) \land (\neg D \lor E)$.

j. $(A \lor B) \land \neg(A \Rightarrow B)$ is satisfiable.

k. $(A \Leftrightarrow B) \land (\neg A \lor B)$ is satisfiable.

l. $(A \Leftrightarrow B) \Leftrightarrow C$ has the same number of models as $(A \Leftrightarrow B)$ for any fixed set of proposition symbols that includes A, B, C.

7.5 Prove each of the following assertions:

a. α is valid if and only if $\mathit{True} \models \alpha$.
b. For any α, $\mathit{False} \models \alpha$.
c. $\alpha \models \beta$ if and only if the sentence $(\alpha \;\Rightarrow\; \beta)$ is valid.
d. $\alpha \equiv \beta$ if and only if the sentence $(\alpha \;\Leftrightarrow\; \beta)$ is valid.
e. $\alpha \models \beta$ if and only if the sentence $(\alpha \wedge \neg\beta)$ is unsatisfiable.

7.6 Prove, or find a counterexample to, each of the following assertions:

a. If $\alpha \models \gamma$ or $\beta \models \gamma$ (or both) then $(\alpha \wedge \beta) \models \gamma$
b. If $\alpha \models (\beta \wedge \gamma)$ then $\alpha \models \beta$ and $\alpha \models \gamma$.
c. If $\alpha \models (\beta \vee \gamma)$ then $\alpha \models \beta$ or $\alpha \models \gamma$ (or both).

7.7 Consider a vocabulary with only four propositions, A, B, C, and D. How many models are there for the following sentences?

a. $B \vee C$.
b. $\neg A \vee \neg B \vee \neg C \vee \neg D$.
c. $(A \;\Rightarrow\; B) \wedge A \wedge \neg B \wedge C \wedge D$.

7.8 We have defined four binary logical connectives.

a. Are there any others that might be useful?
b. How many binary connectives can there be?
c. Why are some of them not very useful?

7.9 Using a method of your choice, verify each of the equivalences in Figure 7.11 (page 249).

7.10 Decide whether each of the following sentences is valid, unsatisfiable, or neither. Verify your decisions using truth tables or the equivalence rules of Figure 7.11 (page 249).

a. $\mathit{Smoke} \;\Rightarrow\; \mathit{Smoke}$
b. $\mathit{Smoke} \;\Rightarrow\; \mathit{Fire}$
c. $(\mathit{Smoke} \;\Rightarrow\; \mathit{Fire}) \;\Rightarrow\; (\neg\mathit{Smoke} \;\Rightarrow\; \neg\mathit{Fire})$
d. $\mathit{Smoke} \vee \mathit{Fire} \vee \neg\mathit{Fire}$
e. $((\mathit{Smoke} \wedge \mathit{Heat}) \;\Rightarrow\; \mathit{Fire}) \;\Leftrightarrow\; ((\mathit{Smoke} \;\Rightarrow\; \mathit{Fire}) \vee (\mathit{Heat} \;\Rightarrow\; \mathit{Fire}))$
f. $(\mathit{Smoke} \;\Rightarrow\; \mathit{Fire}) \;\Rightarrow\; ((\mathit{Smoke} \wedge \mathit{Heat}) \;\Rightarrow\; \mathit{Fire})$
g. $\mathit{Big} \vee \mathit{Dumb} \vee (\mathit{Big} \;\Rightarrow\; \mathit{Dumb})$

7.11 Any propositional logic sentence is logically equivalent to the assertion that each possible world in which it would be false is not the case. From this observation, prove that any sentence can be written in CNF.

7.12 Use resolution to prove the sentence $\neg A \wedge \neg B$ from the clauses in Exercise 7.20.

7.13 This exercise looks into the relationship between clauses and implication sentences.

a. Show that the clause $(\neg P_1 \vee \cdots \vee \neg P_m \vee Q)$ is logically equivalent to the implication sentence $(P_1 \wedge \cdots \wedge P_m) \Rightarrow Q$.

b. Show that every clause (regardless of the number of positive literals) can be written in the form $(P_1 \wedge \cdots \wedge P_m) \Rightarrow (Q_1 \vee \cdots \vee Q_n)$, where the Ps and Qs are proposition symbols. A knowledge base consisting of such sentences is in **implicative normal form** or **Kowalski form** (Kowalski, 1979).

IMPLICATIVE NORMAL FORM

c. Write down the full resolution rule for sentences in implicative normal form.

7.14 According to some political pundits, a person who is radical (R) is electable (E) if he/she is conservative (C), but otherwise is not electable.

a. Which of the following are correct representations of this assertion?

(i) $(R \wedge E) \iff C$

(ii) $R \Rightarrow (E \iff C)$

(iii) $R \Rightarrow ((C \Rightarrow E) \vee \neg E)$

b. Which of the sentences in (a) can be expressed in Horn form?

7.15 This question considers representing satisfiability (SAT) problems as CSPs.

a. Draw the constraint graph corresponding to the SAT problem

$$(\neg X_1 \vee X_2) \wedge (\neg X_2 \vee X_3) \wedge \ldots \wedge (\neg X_{n-1} \vee X_n)$$

for the particular case $n = 5$.

b. How many solutions are there for this general SAT problem as a function of n?

c. Suppose we apply BACKTRACKING-SEARCH (page 215) to find *all* solutions to a SAT CSP of the type given in (a). (To find *all* solutions to a CSP, we simply modify the basic algorithm so it continues searching after each solution is found.) Assume that variables are ordered $X_1, \ldots, X_n$ and *false* is ordered before $true$. How much time will the algorithm take to terminate? (Write an $O(\cdot)$ expression as a function of n.)

d. We know that SAT problems in Horn form can be solved in linear time by forward chaining (unit propagation). We also know that every tree-structured binary CSP with discrete, finite domains can be solved in time linear in the number of variables (Section 6.5). Are these two facts connected? Discuss.

7.16 Explain why every nonempty propositional clause, by itself, is satisfiable. Prove rigorously that every set of five 3-SAT clauses is satisfiable, provided that each clause mentions exactly three distinct variables. What is the smallest set of such clauses that is unsatisfiable? Construct such a set.

7.17 A propositional *2-CNF* expression is a conjunction of clauses, each containing *exactly 2* literals, e.g.,

$$(A \vee B) \wedge (\neg A \vee C) \wedge (\neg B \vee D) \wedge (\neg C \vee G) \wedge (\neg D \vee G) .$$

a. Prove using resolution that the above sentence entails G.

b. Two clauses are *semantically distinct* if they are not logically equivalent. How many semantically distinct 2-CNF clauses can be constructed from n proposition symbols?

c. Using your answer to (b), prove that propositional resolution always terminates in time polynomial in n given a 2-CNF sentence containing no more than n distinct symbols.

d. Explain why your argument in (c) does not apply to 3-CNF.

7.18 Consider the following sentence:

$$[(Food \;\Rightarrow\; Party) \lor (Drinks \;\Rightarrow\; Party)] \;\Rightarrow\; [(Food \land Drinks) \;\Rightarrow\; Party] \,.$$

a. Determine, using enumeration, whether this sentence is valid, satisfiable (but not valid), or unsatisfiable.

b. Convert the left-hand and right-hand sides of the main implication into CNF, showing each step, and explain how the results confirm your answer to (a).

c. Prove your answer to (a) using resolution.

DISJUNCTIVE NORMAL FORM

7.19 A sentence is in **disjunctive normal form** (DNF) if it is the disjunction of conjunctions of literals. For example, the sentence $(A \land B \land \neg C) \lor (\neg A \land C) \lor (B \land \neg C)$ is in DNF.

a. Any propositional logic sentence is logically equivalent to the assertion that some possible world in which it would be true is in fact the case. From this observation, prove that any sentence can be written in DNF.

b. Construct an algorithm that converts any sentence in propositional logic into DNF. (*Hint*: The algorithm is similar to the algorithm for conversion to CNF given in Section 7.5.2.)

c. Construct a simple algorithm that takes as input a sentence in DNF and returns a satisfying assignment if one exists, or reports that no satisfying assignment exists.

d. Apply the algorithms in (b) and (c) to the following set of sentences:

$$\begin{aligned} &A \;\Rightarrow\; B \\ &B \;\Rightarrow\; C \\ &C \;\Rightarrow\; \neg A \,. \end{aligned}$$

e. Since the algorithm in (b) is very similar to the algorithm for conversion to CNF, and since the algorithm in (c) is much simpler than any algorithm for solving a set of sentences in CNF, why is this technique not used in automated reasoning?

7.20 Convert the following set of sentences to clausal form.

S1: $A \;\Leftrightarrow\; (B \lor E)$.
S2: $E \;\Rightarrow\; D$.
S3: $C \land F \;\Rightarrow\; \neg B$.
S4: $E \;\Rightarrow\; B$.
S5: $B \;\Rightarrow\; F$.
S6: $B \;\Rightarrow\; C$

Give a trace of the execution of DPLL on the conjunction of these clauses.

7.21 Is a randomly generated 4-CNF sentence with n symbols and m clauses more or less likely to be solvable than a randomly generated 3-CNF sentence with n symbols and m clauses? Explain.

7.22 Minesweeper, the well-known computer game, is closely related to the wumpus world. A minesweeper world is a rectangular grid of N squares with M invisible mines scattered among them. Any square may be probed by the agent; instant death follows if a mine is probed. Minesweeper indicates the presence of mines by revealing, in each probed square, the *number* of mines that are directly or diagonally adjacent. The goal is to probe every unmined square.

a. Let $X_{i,j}$ be true iff square $[i, j]$ contains a mine. Write down the assertion that exactly two mines are adjacent to [1,1] as a sentence involving some logical combination of $X_{i,j}$ propositions.

b. Generalize your assertion from (a) by explaining how to construct a CNF sentence asserting that k of n neighbors contain mines.

c. Explain precisely how an agent can use DPLL to prove that a given square does (or does not) contain a mine, ignoring the global constraint that there are exactly M mines in all.

d. Suppose that the global constraint is constructed from your method from part (b). How does the number of clauses depend on M and N? Suggest a way to modify DPLL so that the global constraint does not need to be represented explicitly.

e. Are any conclusions derived by the method in part (c) invalidated when the global constraint is taken into account?

f. Give examples of configurations of probe values that induce *long-range dependencies* such that the contents of a given unprobed square would give information about the contents of a far-distant square. (*Hint*: consider an $N \times 1$ board.)

7.23 How long does it take to prove $KB \models \alpha$ using DPLL when α is a literal *already contained in KB*? Explain.

7.24 Trace the behavior of DPLL on the knowledge base in Figure 7.16 when trying to prove Q, and compare this behavior with that of the forward-chaining algorithm.

7.25 Write a successor-state axiom for the *Locked* predicate, which applies to doors, assuming the only actions available are *Lock* and *Unlock*.

7.26 Section 7.7.1 provides some of the successor-state axioms required for the wumpus world. Write down axioms for all remaining fluent symbols.

7.27 Modify the HYBRID-WUMPUS-AGENT to use the 1-CNF logical state estimation method described on page 271. We noted on that page that such an agent will not be able to acquire, maintain, and use more complex beliefs such as the disjunction $P_{3,1} \lor P_{2,2}$. Suggest a method for overcoming this problem by defining additional proposition symbols, and try it out in the wumpus world. Does it improve the performance of the agent?