CHAPTER

20 Conclusion

This book has provided a detailed treatment of commonsense reasoning and a guide to automating commonsense reasoning using predicate logic and the event calculus. We have described general methods that can be applied to a wide range of commonsense reasoning problems. In this chapter, we summarize what has been accomplished and discuss where this work is leading.

20.1 WHAT WAS ACCOMPLISHED?

We started by describing commonsense reasoning, a common activity that people perform almost without realizing it. We then considered several examples of commonsense reasoning and presented important aspects of this activity that need to be treated by any method for automating it or any formalism for describing it. We demonstrated that all these aspects can be handled using a particular logic, the event calculus. We described the event calculus, showed how it can be used for commonsense reasoning using several reasoning programs, and presented some sample applications. We reviewed other logical and nonlogical methods for commonsense reasoning, and we described methods for acquisition of commonsense knowledge.

20.1.1 WHAT IS THE EVENT CALCULUS?

The event calculus is based on classical many-sorted predicate logic with equality. It is characterized by a set of axioms and definitions: 17 in EC and 12 in DEC. The event calculus uses reification: It treats time-varying properties and events as objects so that statements can be made about the truth values of properties at timepoints and the occurrences of events at timepoints.

The event calculus uses circumscription, or the minimization of the extensions of predicates, to allow default reasoning. The event calculus uses the default assumptions that (1) the only events that occur are those known to occur and (2) the only effects of events are those that are known. When new information becomes available, the circumscriptions can be recomputed and reasoning again performed, which may lead to new conclusions. Circumscription is defined by a formula of second-order logic, but in most cases of interest circumscriptions compile into formulas of first-order logic.

An event calculus domain description consists of an axiomatization, observations of world properties, and a narrative of known world events. Given a domain

description, various types of commonsense reasoning can be performed. Temporal projection consists of predicting the results of events given an initial state. Abduction consists of determining what events might have led from an initial state to a final state. Postdiction consists of determining the initial state given events and a final state. Model finding consists of determining properties and events that agree with a given narrative and set of observations.

20.1.2 HOW IS THE EVENT CALCULUS USED?

We have discussed a number of techniques for using the event calculus to perform commonsense reasoning. How do we go about doing this? First, we create an axiomatization that will be used to solve various problems. Second, we create a domain description for any given problem we wish to solve. Third, we solve the problem using manual proof techniques or using automated reasoning programs such as the Discrete Event Calculus Reasoner or answer set programming.

The axiomatization is created as follows. The kinds of problems to be handled are considered and the areas needing formalization are determined; it is useful to work with several sample problems. Sorts, subsort relationships, and variables for each sort are defined. Predicate, function, fluent, and event symbols are defined. The symbols are then used to write the necessary axioms for the areas of interest.

The domain description for a problem is created as follows. The logical constants for the particular problem are defined. These constants are used to specify observations of time-varying properties (fluents) and a narrative of event occurrences. Temporal ordering formulas, which relate the times of properties and events, may be specified. For example, it may be known that one event occurs prior to another event. Unique names axioms for events and fluents are specified.

20.2 WHERE IS THIS LEADING?

What is the reach of the techniques described in this book? Where is this leading? What we have here is the following. Given an axiomatization of some commonsense domains and incomplete information about any given world scenario involving those domains, we can run a reasoning program to fill in missing information about that scenario. This missing information can include events, initial states, intermediate states, and final states. In sum, we have a strategy for building computer systems that understand any commonsense domains of interest. The strategy is to axiomatize the domains, represent a scenario, and then run a reasoning program. We can use this strategy to enrich countless applications with commonsense reasoning capabilities.

There are several potential obstacles to the use of commonsense reasoning. The first obstacle is computing resources. Not all commonsense reasoning problems can be solved efficiently on standard computers. There are several ways of dealing with this. We can use large computer clusters and distributed computing networks. We can continue to perform research on the efficiency of reasoning algorithms in order

to discover and implement more efficient algorithms. We can also use the knowledge gained from this research to state problems in such a way that they can be solved efficiently. For example, experiments have shown that random SAT problems are most quickly solved either when they are underconstrained so that they have many solutions, or when they are overconstrained so that they have few solutions.

Another potential obstacle is our ability to construct complete and correct axiomatizations. Although we have presented appropriate axiomatizations and techniques for a number of commonsense areas, the commonsense world is large. Considerable effort and ingenuity are required to develop good axiomatizations. To some degree the task can be eased by using acquisition techniques such as crowdsourcing and text mining.

Another potential obstacle is that of linking up the event calculus to the world. How do we transform a problem in the world into an event calculus problem? Certain problems such as those involving electronic payment systems use representations that are easily translated into the event calculus. Other problems require more extensive preprocessing. Natural language text or speech must first be processed by syntactic and semantic parsers, and images must first be treated by image processing software.

20.3 CLOSING REMARKS

This book has considered commonsense reasoning and how the event calculus can capture it. This formalization of commonsense reasoning has two main benefits: It brings us a greater understanding of the human capacity for commonsense reasoning and it allows us to build more flexible and friendly computer systems. The current situation of automated commonsense reasoning can be likened to the situation of the web in the early 1990s. Decades of research on hypertext and computer networking had finally crystallized into two simple and powerful mechanisms for sharing information online: the hypertext transfer protocol (HTTP) and the hypertext markup language (HTML). In the years following, these mechanisms allowed the online world to take off exponentially. Similarly, decades of research have now crystallized into a simple and powerful mechanism for commonsense reasoning, the event calculus. I hope that the event calculus will lead to an explosion of applications. I invite you to build them.

BIBLIOGRAPHIC NOTES

Mitchell, Selman, and Levesque (1992) study the time taken by the Davis-Putnam-Logemann-Loveland procedure (M. Davis, Logemann, & Loveland, 1962; M. Davis & Putnam, 1960) to solve randomly generated 3-SAT problems, or SAT problems whose clauses contain three literals. They vary the ratio of the number of clauses to the number of variables and find that problems with a low-clause-to-variable ratio (around 2) or a high-clause-to-variable ratio (around 8) are quickly solved. The problems with a low-clause-to-variable ratio are underconstrained—most truth

assignments are satisfying. The problems with a high-clause-to-variable ratio are overconstrained—few if any truth assignments are satisfying. On the other hand, problems with a clause-to-variable ratio around 4.25, for which approximately half the truth assignments are satisfying, take much longer to solve. Baum (2004, pp. 22, 23, 300-302) argues that the world is overconstrained, which allows humans to develop compact and overconstrained representations, which in turn allows reasoning to proceed efficiently.