COMPUTATION AND ACTION
UNDER BOUNDED RESOURCES

A DISSERTATION
SUBMITTED TO THE PROGRAM IN MEDICAL INFORMATION SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

By
Eric Joel Horvitz
December 1990

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Ronald A. Howard
Department of Engineering–Economic Systems
Stanford University
(Principal Advisor)

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

Edward H. Shortliffe
Departments of Medicine and Computer Science
Stanford University

I certify that I have read this dissertation and that in my opinion it is fully adequate, in scope and in quality, as a dissertation for the degree of Doctor of Philosophy.

George B. Dantzig
Department of Operations Research

Approved for the University Committee on Graduate Studies:

Dean of Graduate Studies

Abstract

I define and implement a model of rational action for automated reasoning systems that makes use of flexible approximation methods and decision-theoretic procedures to determine how best to solve a problem under bounded computational resources. The model provides a perspective on the use of metareasoning techniques to balance the costs of increased delays with the benefits of better results in a decision context. I focus on the use of inexpensive real-time analyses to control the allocation of computational resources in complex decision-theoretic reasoning. The approach extends traditional decision analyses to autoepistemic models that represent knowledge about problem solving, in addition to knowledge about distinctions and relationships in the world.

To investigate the use of decision analysis for controlling computation, I constructed a computer program named Protos. Protos uses information about the progress of problem solving to identify the ideal time to halt computation and take action in the world. Protos' metareasoner controls the precision of probabilities inferred from complex network models that represent domain-specific expertise about uncertain relationships among observations and hypotheses. I found that it can be valuable to allocate a portion of costly reasoning resources to deliberate about the best way to solve a decision problem. In addition to serving as a testbed for exploring the value of metareasoning, I made use of Protos to examine the integration of reflex and deliberative analyses and the construction of time-dependent utility models from

observations.

After discussing principles for applying multiattribute utility theory to the control of basic computational procedures, I describe how these principles can be used to control probabilistic reasoning. In particular, I present techniques for controlling, at run time, the tradeoff between the complexity of detailed, accurate analyses and the tractability of less complex, yet less accurate probabilistic inference. Then, I describe the architecture and functionality of Protos and review the system's behavior on high-stakes decision problems in medicine. Finally, I move beyond the consideration of time constraints to investigate the constraints on decision-theoretic reasoning posed by the cognitive limitations of people seeking insight from automated decision systems.

Guide for the Reader

In this dissertation, we explore theoretical and empirical work on rational decision making that addresses topics of interest to investigators in computer science, decision analysis, and medical informatics. Essential background on the historical and theoretical context of this dissertation is presented in Chapter 1. Readers who are unfamiliar with the theoretical foundations of decision theory, or with the use of belief networks and influence diagrams for representing and reasoning with uncertain knowledge, will benefit from the background on probability and decision theory provided in Appendix A.

Basic concepts and definitions that are useful for the analysis of problems of ideal approximation under bounded resources are found in Chapter 2. Chapter 3 contains illustrative examples of the multiattribute nature of partial results that are generated by sorting algorithms. The examples demonstrate how the ideal computational strategy to select can be sensitive to the preferences of a user. This work may be of special interest to computer scientists who have investigated primarily the worst-case complexity of problem solving. Readers with little interest in the utility analysis of partial results generated by computational algorithms can skip this chapter.

In Chapter 4, the principles elucidated in Chapter 2 are extended to problems of belief and action under uncertainty. We examine closed-form equations that report a measure of the value of computation for probability-bounding algorithms. Chapter 5 contains a discussion of a flexible algorithm for probabilistic inference. The chapter includes a discussion of the algorithm's properties and limitations. Readers with little interest in the details of this bounding algorithm may wish to skip this chapter. Chapter 6 contains details about the implementation of the Protos system. We review the architecture of Protos and explore the construction and representation of time-dependent utility models. The chapter also includes a presentation of the range of Protos' behavior, illustrated with samples of the system's output. Readers should, at minimum, peruse the output of the Protos system to build intuitions about the costs and benefits of continuing to compute in a time-pressured setting. Chapter 7 contains case analyses of the performance of Protos on several different belief networks in the context of decision problems in critical-care medicine. The chapter contains discussion of an evaluation technique that allows us to probe the value of Protos' metareasoning, and describes the output of Protos' case-analysis summarizer.

In Chapter 8, we shall the generalize methods for considering the value of computation under time constraints to the optimization of computer-based decision analysis under the constraints defined by the cognitive limitations of people using decision-theoretic systems. The chapter contains a discussion of how we can make advice generated by a computer system more valuable to users by introducing flexibility into value-of-information inference. The approach allows us to simplify the procedures so that they are more natural and understandable. Readers who are primarily interested in the comprehensibility, naturalness, and explainability of decision-theoretic inference may wish to read Chapter 8 in isolation from the other dissertation chapters. Finally, Chapter 9 contains a summary of this dissertation research and discusses promising extensions of the work on decision-theoretic metareasoning.

Acknowledgments

I am grateful to many friends and colleagues for their enthusiastic support of my research. Ron Howard, who served as my principle dissertation advisor, had a significant effect on the course of my research. I have been inspired by Ron's intuitions and his ability to see immediately the sensitive variables in a problem. I have also been moved by Ron's efforts to communicate his ideas beyond the university, to bring insight to decisions in personal and professional arenas. Throughout my doctoral work, Ted Shortliffe has provided insightful technical advice and research intuitions on automated problem solving. I have valued his perspective on problems and opportunities with the use of computers to enhance medical care. Ted's creation of the unique Medical Computer Science Group, within the Knowledge Systems Laboratory at Stanford, provided me with the freedom and resources to pursue my interdisciplinary interests. George Dantzig never failed to cut through abstract issues with straightforward, focusing questions. George's broad interests in computation—from problems in algorithmic complexity to the generation of plans under uncertainty—cast additional light on the many bridges that span the historical boundaries between operations research, computer science, and artificial intelligence.

Beyond my principal advisors, David Heckerman has been a close friend and fellow-traveler in the study of intelligent problem solving. Our countless discussions and technical collaboration have led to a few answers, and to a greater number of new questions that undoubtedly will continue to shape our research agendas. Greg Cooper provided ongoing feedback, creative suggestions, and detailed technical advice on my dissertation research. Pat Suppes was an early supporter of my attempt to formalize rational action under computational resource constraints. He has continued to be a role model with his deep thinking and passion for excellence in tackling difficult problems in the physical and behavioral sciences.

I thank Jaap Suermondt for exciting collaboration that led to the development of the bounded-conditioning algorithm. I am indebted to Geoff Rutledge for providing me with a fountain of clinical expertise on time-pressured decision making in medicine. Jack Breese provided useful feedback on the concepts and presentation of the ideas. Doug Ellison, Bharat Nathwani, and Keung-Chi Ng provided assistance with the assessment and use of abstraction hierarchies in pathology diagnosis. Other friends and advisors in computer science, decision analysis, and medical informatics who provided me with valuable suggestions and feedback include Bruce Buchanan, Jon Doyle, Brad Efron, Brad Farr, Ed Feigenbaum, Jack Good, Barbara Hayes-Roth, Max Henrion, Holly Jimison, Harold Lehmann, Randy Miller, Nils Nilsson, Howard Pattee, Ramesh Patil, Judea Pearl, Mark Peot, Ross Shachter, Peter Szolovits, and Michael Wellman. I am grateful to Lyn Dupré for providing editorial assistance with this dissertation. I thank Cynthia for bearing with the eternal whir of the computer and with my incessant chatter about probability, problem solving, and metareasoning. Finally, I am deeply indebted to my family, particularly to my parents, for all they have given me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The nature of rational action has been debated for centuries. A familiar position, taken by a great number of researchers in the behavioral and decision sciences, is that rational decisions are those that maximize a numerical measure of preference, termed *utility*. Utility is defined in decision theory by the axioms of utility, developed by von Neumann and Morgenstern over four decades ago (von Neumann and Morgenstern, 1947). Decision theory was developed to help people reason about taking action under *uncertainty*. The inescapable incompleteness in our knowledge about the world leads to unavoidable uncertainties about the consequences that our actions will have.

Early pioneers of symbol-processing models of intelligence dismissed decision theory for use in automated reasoning systems. Although many of these investigators viewed decision theory as a gold standard for action in the presence of adequate representational and computational resources, the formal methodology was rejected as too complex for reasoning under bounded resources. Citing the limited abilities of human decision makers and the forgiving nature of many problems in the world, researchers proposed that most intelligent behavior is oriented toward finding solutions that are nonoptimal, yet are sufficient or *satisficing* (Simon, 1955). A primary task in modeling intelligent behavior was viewed as devising logical procedures that could forward the goals of an intelligent decision maker—and not necessarily as optimizing the value of reasoning or remaining consistent with decision theory.

To date, research on automated reasoning, performed under the label of *bounded rationality*, has yielded systems whose behavior may stray far from the levels of utility that might be achieved through more sophisticated decision-theoretic analyses. Losses may be especially significant in high-stakes decision making, given complex uncertainties about the world. For example, bypassing principled approaches in the design of medical reasoning systems that are used to assist physicians with diagnosis or to control therapeutic devices autonomously may be costly to patients. Such potential losses—and opportunities for great gain—highlight the possible usefulness of more sophisticated decision-theoretic analysis for optimizing the value of behavior under resource constraints.

Although the straightforward solution of decision-theoretic models can generate "optimal" recommendations for action in a timeless world, the delay required for solving complex models can lead to a poor outcome. A decision maker typically incurs a loss in the value of his action in the world by waiting to address a challenge. The cost of delay is based in irrevocable losses in opportunity stemming from such universal time-dependent processes as (1) competition for limited resources, (2) decay of physiologic states, and (3) coordination among independent decision makers.

I shall describe my work on the synthesis of artificial intelligence (AI) and decision analysis (DA) to address the problem of ideal computer-based decision making under varying computational resource constraints. I have investigated principled techniques for making rational decisions under resource constraints. Rather than reject the pursuit of a theoretical foundation for computing rational belief and action, I have sought to extend coherently the principles of normative rationality to situations of uncertain, varying, and scarce reasoning resources. My thesis is that we can use a principled theory of action to build computer-based reasoners for making recommendations or for taking autonomous action under scarce and varying resources.

We shall examine the extension of traditional decision-theoretic analyses with techniques that enable a system to consider the value and cost of reasoning under limited resources. A model of rational action for automated reasoning systems shall be developed that makes use of flexible approximation methods and inexpensive decision-theoretic procedures to decide how best to solve a problem given costs or constraints on reasoning resources. Flexible approximation strategies give reasoning systems the ability to trade off gracefully optimality for increased tractability. Procedures for controlling flexible reasoning can be generated dynamically in tractable real-time analyses, performed in conjunction with problem solving, or can be derived in off line decision analyses.

I develop real-time analyses to control decision-theoretic reasoning at the base-level. This approach hinges on the extension of traditional decision analyses to autoepistemic models that represent knowledge about problem solving, in addition to the traditional focus on distinctions and relationships in the world. As we shall see, it can be valuable to allocate a portion of costly reasoning resources to metalevel deliberation about the best way to use additional resources to solve a decision problem.

1.1 Artificial Intelligence and Decision Analysis

For almost 3 decades, AI investigators have endeavored to develop computer-based representations and reasoning strategies for encoding and manipulating knowledge efficiently. AI evolved concurrently with other disciplines, spurred by the development and refinement of electronic computing techniques in the mid-1950s. AI investigators became interested in the metaphor of intelligence as information processing, and early on, distinguished their work from ongoing studies of computer-based information processing in operations research, systems science, and control theory. The researchers turned away from prevalent quantitative models, laden with numeric information, and instead focused on the logical processing of symbolic knowledge (Newell and Simon, 1963; Feigenbaum, 1964; Simon, 1969; McCarthy and Hayes, 1969; Simon, 1972).

AI researchers have had diverse goals. Expert-system developers, or knowledge engineers, have created computer programs that can serve as consultants to professionals in specific disciplines and subdisciplines. Other AI investigators have elucidated basic principles of knowledge representation and reasoning. These researchers have developed fundamental techniques, and, at times, philosophical frameworks, for building intelligent artifacts. Still other AI investigators have used computer models as tools for understanding human cognition. Although AI scientists have had diverse research agendas, they have been unified by an interest in the foundations of intelligent behavior, and by their concerns with computational and representational tractability. Thus, many AI researchers have rejected formal models traditionally associated with intractable solutions. Instead, they have opted to test empirically the efficacy of informal heuristic reasoning procedures.

Decision analysis (DA) evolved during the same period as did AI (Howard, 1966; Howard, 1988; Raiffa, 1968; Keeney and Raiffa, 1976). Decision analysts share with many AI scientists interests in the representation of abstract knowledge, in fostering tractable analyses of decision problems, and in intelligent decision making. Unlike AI, decision analysis evolved with a commitment to applying the principles of decision theory. Also, almost all work on decision analysis has centered on consultation rather than on computer-based generation of recommendations; decision analysts have traditionally performed custom analyses for clients to bring clarity to important decisions. A decision analysis includes a careful assessment of the desires or *preferences* of a decision maker, the alternative actions available to that decision maker, and the outcomes of these actions and their associated uncertainties. In practice, decision analysts use a battery of heuristic and formal techniques to build and to refine iteratively a decision model, and apply decision theory to identify the best action to take under uncertainty. Although most AI investigators are familiar with decision theory, decision-theoretic techniques, and, more recently, advances in decision analysis, have been largely overlooked by that community.

1.2 Paradigms for Reasoning Under Uncertainty

There have been several competing paradigms for research on computer-based reasoning. Many AI investigators consider only logical relationships in their research. Many of these investigators make an implicit assumption that uncertainty about the environment—or about the effect of an action—is minor compared with the difficulties of developing a satisfactory logical solution. Other logicists have devised deterministic machinery for addressing uncertainty. These techniques include nonmonotonic logics and default reasoning (Ginsberg, 1987). In contrast to this group, many researchers build systems that wrestle explicitly with numeric representations of uncertainty, or with logical methods that are founded on numeric representations of belief (Nilsson, 1986). Investigators who admit the use of real-numbered degrees of truth for reasoning under uncertainty can be subdivided into two groups: those who investigate the use of formal reasoning techniques, and those who study informal, heuristic reasoning techniques.

Formalists study systems based on small sets of axioms that define consistent theories of reasoning. They hope that reasoning systems based on such principles will explain and generate complex intelligent behaviors—much as the parsimonious set of astronomical laws developed by Kepler can explain the complex motion of heavenly bodies. AI investigators have explored several axiom-based methodologies for reasoning under uncertainty. These include probability theory, Dempster–Shafer theory (Shafer, 1976), fuzzy-set theory (Zadeh, 1983), and multivalued logics (Gaines, 1978). All these approaches are theories for assigning measures of partial truth or *belief* to alternate hypotheses, given observations or evidence. The theories do not address rules for *taking action* under uncertainty. Decision theory is a formal calculus for determining the best action under uncertainty. Decision theory is defined by the axioms of utility, which, in turn depends on probability. Thus, we can consider decision theory to be the axioms of probability and utility.

Many people find the axioms of probability and utility persuasive as principles for rational choice under uncertainty (see Appendix A for additional discussion on the axioms of probability and utility). Decision theory has been accepted as a foundation for reasoning about belief and action in a wide range of fields, from economics to behavioral science. Although *normative* can be used to describe the behavior entailed by any compelling set of consistent principles for reasoning about action,[1] the term is most frequently used to refer to reasoning about belief and action in accordance with decision-theoretic principles. We shall use *normative* to refer to decision theory.

[1]For example, an investigator assuming a deterministic world for an AI research problem might consider logical inference to be "normative" in that logic prescribes a set of rules for correct inference under certainty.

Researchers interested in normative reasoning methods study techniques for building reasoners that perform consistently with principles of decision theory for determining action.

AI investigators interested in heuristic methods for reasoning under uncertainty reject the requirement for a set of mathematically specified principles and for well-defined behavior, often noting problems with limited computational resources, but equally often arguing that formal methods such as decision-theoretic techniques lack the expressiveness needed for intelligent behavior. Instead of relying on axioms, these researchers seek to solve problems with the development of tractable, intuitive approaches. Many AI investigators view heuristics as an approach to solving difficult reasoning problems that is more direct and tractable than is working within the constraints and combinatorics of decision theory. Heuristic approaches often are *descriptive* in that they attempt to describe human problem solving—much as the epicycle machines, used in the days before we had a satisfying theory of planetary motion, could describe adequately the motions of the stars. Scientists often draw insights about heuristic procedures by introspecting or by observing people solving problems.

1.3 Normative Reasoning in Intelligent Systems

Although decision theory was well known to early AI scientists, it was rejected as inappropriate for intelligent problem solving. Herbert Simon was an early proponent of research on heuristic reasoning, having noted problems with the straightforward application of decision theory. In 1955, he observed that, although decision theory defines rational action in a world of unbounded computation resources, we should consider constraints on cognitive resources in generating and evaluating the behavior of real-world decision makers (Simon, 1955). Simon used *bounded rationality* to refer to the importance of acceptable or *satisficing* decision-making behavior. Simon's discussions stimulated research on heuristic approaches to decision making in a variety of disciplines. Economists and business analysts studied related issues of bounded rationality for characterizing the decision-making behavior of populations (March, 1978; Shugan, 1980).

1.3.1 Early Research in Medical Informatics

Despite the dominant interests of AI investigators on logical reasoning and heuristic models of problem solving, research on the automation of normative reasoning progressed in the late 1950s. In particular, early research projects on the automation of medical reasoning centered on the implementation of normative expert reasoners (Ledley and Lusted, 1959). In the early 1960s, several teams of physicians and computer scientists began to experiment with computer-based applications of decision-theoretic reasoning. These early medical-informatics pioneers worked on computer programs with the hope that more complex normative reasoning systems might some day help physicians to solve difficult problems. Prototype probabilistic and decision-theoretic medical diagnostic systems were constructed, including Warner's system for the diagnosis of heart disease (Warner et al., 1961), de Dombal's system for the analysis of acute abdominal pain (de Dombal et al., 1972), and Gorry's systems for heart disease (Gorry and Barnett, 1968) and renal failure (Gorry, 1973). To maintain the tractability of knowledge representation and inference procedures, the investigators engineered systems for reasoning in small, well-circumscribed problem areas, and imposed the simplifying assumptions of mutual exclusivity among diseases (only one disease can be present) and conditional independence among all symptoms. Nevertheless, the systems were found to perform at a level comparable to experts within the scope of their limited domains (Gorry, 1973; de Dombal et al., 1974; Dawes and Corrigan, 1974).

1.3.2 Normative Methods as Intractable and Opaque

In the early 1970s, medical-informatics investigators began to dream of building medical reasoning systems with broader abilities and a greater breadth of knowledge than the earlier normative systems had. Problems with the tractability and expressiveness of normative reasoning and representation became salient as investigators attempted to scale up their systems to handle real-world patient cases. Scientists highlighted the complexity of constructing and manipulating large knowledge bases of probabilistic information, discussed the inadequacy of making assumptions of conditional independence, and of mutual exclusivity and exhaustivity of hypotheses, to gain tractability, and expressed concern that a combinatorial explosion would threaten attempts to move beyond these assumptions or to larger domains (Gorry, 1973; Szolovits, 1982). Beyond problems with representational and computational intractability, some researchers pointed out the limited expressiveness of normative representations, citing the apparent differences between the quantitative approach of probabilistic inference and the richer, qualitative nature of informal human reasoning. Some suggested that the differences between normative methods and informal human reasoning could lead to problems in encoding expertise and in explaining the results of probabilistic inference (Gorry, 1973; Shortliffe and Buchanan, 1975; Szolovits, 1982; Davis, 1982).

Frustration with the complexity of the task of collecting, representing, and reasoning with large amounts of probabilistic information stimulated interest in the value of nonnormative, heuristic methods for reasoning under uncertainty. Gorry, a pioneer in the application of normative reasoning in medicine, captured the growing disillusionment with probability and utility—and growing interest in informal reasoning strategies—with his comment in a landmark paper:

> Although we cannot characterize precisely the methods used by experts, it is clear that these methods can accommodate the greater complexity of real clinical situations. (Gorry, 1973, p. 49)

Some medical-informatics researchers became interested in the logical reasoning procedures developed in the maturing discipline of AI. At the same time, AI investigators began in earnest to apply AI representation and reasoning techniques to build expert systems in medicine. The ubiquity of uncertainty in medicine catalyzed the development of heuristic *scoring* methodologies for assigning belief to hypotheses and the integration of logical reasoning techniques with uncertainty calculi.

1.3.3 Interest in Heuristic Approaches

Several AI in medicine projects formulated and studied nonprobabilistic numeric approaches to reasoning under uncertainty. The Internist-1 project (Miller et al., 1982) was an early project to develop and use an ad hoc scoring scheme for assigning a measure of likelihood to diseases in internal medicine, given observations.[2] The investigators acquired several different numeric measures and used these measures to assign belief to different diseases. The Internist-1 team also created several heuristic decision-making methods to generate recommendations about new evidence to acquire and tests to perform, given the current belief in alternate hypotheses.

Another well-known heuristic approach developed in the 1970s was a quasiprobabilistic scheme named *certainty factors*. Certainty factors provided a means of managing belief within rule-based production systems (Buchanan and Shortliffe, 1984; Clancey, 1985). The development was a key component of Mycin (Shortliffe and Buchanan, 1975), a system that applied logic-based reasoning techniques—originally developed by AI investigators for theorem proving—to medical diagnosis and therapy.

Other heuristic approaches for reasoning under uncertainty included the use of patterns of evidential criteria, or *frames*, as in the AI-Rheum system and its descendants (Lindberg et al., 1980), the development of descriptive cognitive constructs, such as *long-term* and *active memory* in the Present-Illness Program (Pauker et al., 1976), and the use of causal networks to relate pathophysiology and observations, as in the Casnet system (Weiss et al., 1978).

Medical-informatics investigators of the 1970s embraced heuristic schemes partly to avoid the unrealistic assumptions of independence they had to make with normative reasoning. Recent analyses have shown that the researchers did not avoid the assumptions by moving out of the probabilistic framework—the assumptions merely became more difficult to identify when heuristic approaches were used. Analyses of several of the heuristic procedures for performing diagnosis and decision making under uncertainty have demonstrated that the methods are as restrictive, or *more restrictive*, in their imposition of independence than were the simple probabilistic systems of the 1960s (Heckerman, 1986; Heckerman and Horvitz, 1987). However, unlike the explicit independence assumptions of the probabilistic systems, the restrictive assumptions in the heuristic approaches have been less apparent.

[2]Internist-1 research has evolved into the current Quick Medical Reference (QMR) (Miller et al., 1986), Caduceus (Pople, 1982), and QMR-DT (Shwe et al., 1990a) projects.

1.3.4 Recent Research at the Boundary of AI and DA

Over the last two decades, several projects have made use of decision theory to address difficult AI problems (Jacobs and Keifer, 1973; Coles et al., 1975; Feldman and Sproull, 1975). In the last five years, however, investigators in a subdiscipline of AI studying techniques for reasoning under uncertainty, have attempted to synthesize AI and DA reasoning and representation techniques (Cooper, 1984; Holtzman, 1985; Smith, 1986; Pearl, 1988; Breese, 1987; Wellman, 1988; Langlotz et al., 1988; Henrion, 1988; Klein, 1989; Heckerman, 1990b; Horvitz et al., 1989a). Most of this research has made use of a graphical knowledge-representation language developed by decision analysts, called *influence diagrams* (Howard and Matheson, 1981; Owen, 1978; Olmsted, 1983). The influence diagram was developed by decision analysts at Stanford Research Institute (SRI) in the early 1970s, partly in response to advances in electronic computing. Although influence diagrams formally describe a decision model, they have a human-oriented qualitative structure that facilitates knowledge acquisition and communication. An influence diagram is a graphical knowledge representation that captures information about decisions, outcomes, and the probabilistic dependencies among propositions and events (Rousseau, 1968; Cooper, 1984; Pearl and Verma, 1987). The representation is, more specifically, a directed acyclic graph (dag) that contains nodes representing uncertain variables, and arcs that represent dependencies among the variables. Arcs missing between nodes in a belief network are significant assertions of conditional independence. Influence diagrams can be used to describe completely problems with inference and action under uncertainty. A closely related representation, called *belief networks*, (also referred to as *knowledge maps* and *probabilistic networks*) is a specialization of influence diagrams that does not include preference and decision information. (See Appendix A for a detailed discussion of belief networks and influence diagrams).

A belief network and its associated probabilities determine a complete joint probability distribution for the problem area represented. Given the joint distribution, we can compute the posterior probability of all hypotheses given any set of observations. Influence diagrams and belief networks have allowed investigators to examine the complexity of probabilistic inference, and to design techniques to take advantage of independence in the world. In 1987, Cooper confirmed intuitions about the complexity of probabilistic inference by showing that inference within belief networks is in the $\mathcal{NP}$-hard class of problems (Cooper, 1990b). Almost all investigators working with influence diagrams have attempted to generalize the simplified inference models of the 1960s by identifying and introducing dependency among variables, and by countering combinatorial explosion by exploiting conditional independence.

Large belief networks have been constructed by AI researchers for computer-based inference about medical diagnosis (Heckerman et al., 1990; Beinlich et al., 1989; Andreassen et al., 1987), robot navigation (Dean and Kanazawa, 1988) and the comprehension of stories (Charniak and Goldman, 1989). Several investigators have developed approximation methods and specialized exact algorithms designed for processing belief networks of different topologies (Pearl, 1986; Pearl, 1988; Shachter, 1986; Lauritzen and Spiegelhalter, 1988; Henrion, 1988). Other investigators have developed techniques for identifying independence efficiently at knowledge-acquisition time (Heckerman, 1990b; Heckerman et al., 1990).

1.4 A Normative-Metareasoning Perspective

The use of belief networks to represent and exploit conditional independence has allowed investigators to build large knowledge bases of probabilistic information, and to increase the efficiency of engineering normative systems. Nevertheless, we still must grapple with the possible great complexity of normative reasoning. Such complexity is highlighted by Cooper's analysis of the worst-case run times for belief-network inference, and by our recent experience with large belief networks (Shwe et al., 1990a). There is little hope that we will be able consistently to avoid potentially costly delays when we solve large normative models in time-pressured settings.

I have explored *normative metareasoning*—the use of decision analysis at the metalevel, to make decisions *about* the nature and extent of problem solving (Horvitz, 1987c; Horvitz, 1988; Horvitz et al., 1989c). Key components of normative metareasoning addressed in this dissertation are (1) the development of new flexible inference strategies that allow us to reason about a continuous spectrum of the completeness of an analysis, and (2) the optimization of the *value* of approximate inference under constraints in computational resource. I have sought to optimize the value of normative reasoning by developing and intelligently controlling flexible approximation strategies. I shall show that we can use normative principles to increase the value of a normative reasoning system, and that these techniques have promise for maximizing the value of a variety of computational procedures. Normative analyses at the metalevel have the ability to take into consideration uncertain knowledge about the cost of reasoning and about problem-solving efficacy.

1.4.1 Reflective Decision-Analytic Models

We can apply normative metareasoning to make decisions about the best way to solve a variety of computational problems. In this dissertation, we shall examine the use of normative metareasoning to guide normative reasoning of greater complexity at the base level. In particular, we shall explore the ideal control of approximate inference strategies in belief networks. I use *reflective decision-analysis* to refer to the use of normative metareasoning for guiding base-level decision-theoretic reasoning. From the DA perspective, reflective decision-analytic models are richer than traditional decision models, in that they consider autoepistemic information about the *process of problem solving* in addition to distinctions and relationships in the world. Autoepistemic distinctions include predicates that represent knowledge about the value of continuing to compute with a particular inference strategy, about the cost of resources consumed by a deliberation step, and about the value and costs of the mechanisms used for metareasoning. We take the use of normative procedures to control base-level normative reasoning as a model of rationality under resource constraints.

Figure 1.1 highlights different classes of knowledge that we can apply in a reflective decision analysis. We can use probability distributions to represent the uncertainty in the costs and efficacies of different reasoning strategies, and of expected challenges from the environment. Other factors include uncertain knowledge about what a decision maker's preferences are and how utilities assigned to states will be weighted and combined in the future. I use *bounded optimality* to refer to the optimization of the value of actions taken by a reasoning system, given assertions about the system's reasoning abilities, the problems expected in an environment, and costs and constraints on reasoning resources. That I have pursued the optimization of reasoning under bounded resources does not necessarily mean that verifiable bounded optimality is my end-goal. Rather, I have sought to make explicit the relative performance of alternative reasoning methodologies, and to generate additional insight about how we might further enhance the activity of our computational systems.

1.4.2 Naive Versus Reflective Normative Systems

In many contexts, the intuitions of AI investigators on the inapplicability of decision-theoretic reasoning strategies can be supported by normative meta-analyses. Given constraints on time or on engineering resources, the solution of a problem with an approximation or more ill-characterized heuristic technique can have a *higher expected value* than does a detailed normative analysis—in spite of ensured suboptimality or uncertainty in the performance of the strategy. In the case of a high-stakes decision problem, such as determining whether a patient in an intensive-care unit should be treated for left-ventricular failure or hypovolemia, or whether a power plant should be shut down, given evidence about a possible breach of coolant, we may not be able to wait while a complex probabilistic model is evaluated. The ideal model or inference strategy to use depends on the cost and availability of time for computation. We wish to endow reflective normative decision systems with such knowledge.

We shall view rational decisions as actions dictated by the *principle of maximum expected utility*: A reasoning system should choose an action (whether it be a problem-solving action or an action in the world) that optimizes the expected utility of a system user or client. From this perspective, committing to an involved decision analysis in time-pressured situations can be *irrational*. It is easy to find examples where the use of systems based on single-level or "naive" normative reasoning models may be irrational. For example, a normative meta-analysis of a heuristic rule that directs a rapid "reflex" response in a time-pressured setting might demonstrate that use of the heuristic has a higher expected utility than would be assigned to a real-time decision analysis of the same problem. The analysis might further reveal that, although the heuristic frequently results in inappropriate action, the long-term losses associated with these inaccuracies are trivial compared with the costs incurred when the normative method misses a deadline for response. In other contexts, the same normative model could be more valuable than the heuristic strategy. Theoretical or empirical analyses of the value of the alternative methods, or of different approximate inference strategies for normative reasoning, could yield rules that we could apply to select the procedure with the greatest expected utility.

To date, normative computer-based inference procedures and representations have not included an explicit consideration of the cost of reasoning. We have few computational tools for addressing the naivety of complex analyses, given limitations in computational or engineering resources. We also have few techniques that could allow a reasoning system to react to *variation* in the problem difficulty and in the time criticality of taking action. Potential benefits of the pursuit of optimal normative reasoning under resource constraints include the construction of reasoning systems that can custom-tailor their response to different contexts dynamically, and that have greater expected value than do simple policies or heuristic approaches. Perhaps more important, the study of normative metareasoning can provide insight about principles of reasoning under scarce resources.

1.4.3 Real-Time Versus Design-Time Metareasoning

Normative metareasoning includes the use of decision-theoretic procedures to reason about alternative problem-solving procedures strategies in an *offline* engineering setting, and in a *real-time* setting. Offline analyses can be useful for developing a priori computational policies that are tuned to an expected set of problem instances. These a priori policies include simple default control rules, or may be more sophisticated schemata for the control of computation designed to react, in real-time, to simple patterns of evidence about a problem instance. Offline normative analyses may indicate that the best real-time strategies for solving a set of problems should

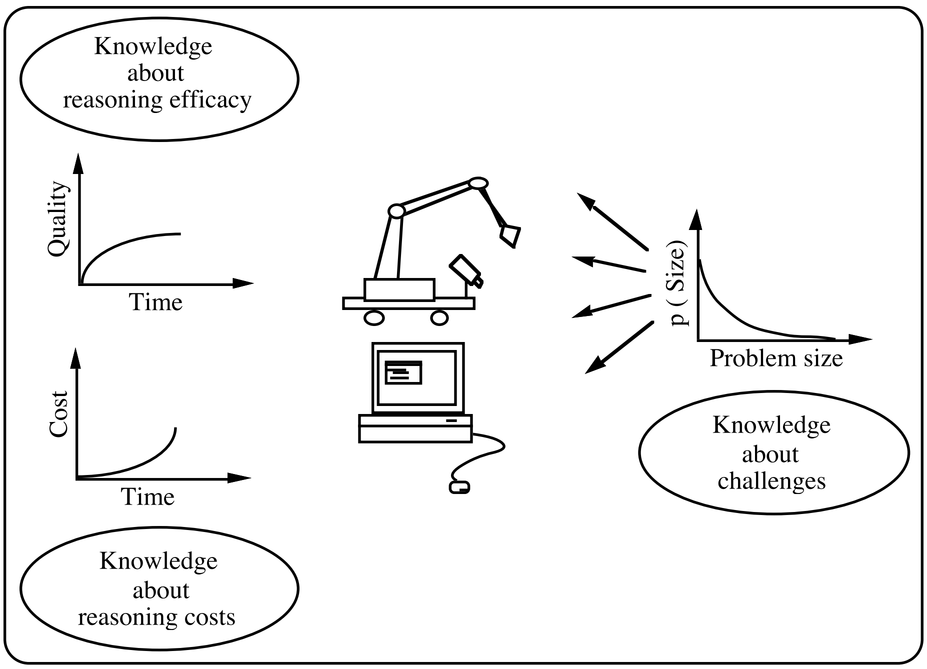


Figure 1.1: Knowledge used in normative metareasoning.
Normative metareasoning systems consider the costs of reasoning and the efficacy of alternative solutions. In the engineering of reasoning systems, we can also consider the frequency of different classes of problems, to create reasoning policies that are ideal for use over time. These classes of knowledge may be uncertain, as indicated by the probability density function representing uncertainty in problem size. As indicated by the icons, techniques for optimizing reasoning under resource constraints can be valuable for decision-support systems, as well as for autonomous reasoning systems.

not rely on the explicit representation and manipulation of probability and utility information. Alternatively, an offline analysis can suggest that real-time normative metareasoning is valuable for solving a class of problems. If an analytic regress is to be avoided, real-time decision making must be far less expensive than base-level inference (we shall touch on analytic-regress issues in Chapters 5 and 9). We wish to identify inexpensive meta-analyses to direct expensive computational processes at the base level. Thus, real-time metareasoning *typically depends on the development of metalevel decision analyses with tractable solutions or approximations.* We shall explore the real-time control of approximate probabilistic reasoning strategies with inexpensive closed-form solutions. Besides time constraints, we may also consider other classes of resource in an offline meta-analysis. Other constraints include the cognitive resources required by a human to understand the conclusions of complex reasoning, and the work required in the knowledge-acquisition phase of building an expert system.

1.5 Flexibility Through Graceful Degradation

We could apply normative metareasoning to compare the value of a large set of normative and nonnormative approaches to reasoning under uncertainty. I have focused my attention on normative approximation methods. Traditional approaches to automated decision-theoretic reasoning and representation have been inflexible. That is, the reasoning methodologies have been directed at performing a complete analysis at a single level of detail. People in many professions are familiar with the *80–20 rule*. This rule is invoked often to describe the sense that, in many problem-solving arenas, it is possible to solve a great portion of a problem of a problem for a small fraction of the effort required to do a complete or perfect analysis. Despite our commonsense knowledge about the typical relationship between the effort expended on an analysis and the quality of a solution, computer scientists have offered few formal methodologies for making such tradeoffs. Instead, a great majority of research on computational problem solving has dwelled on all-or-nothing strategies that provide

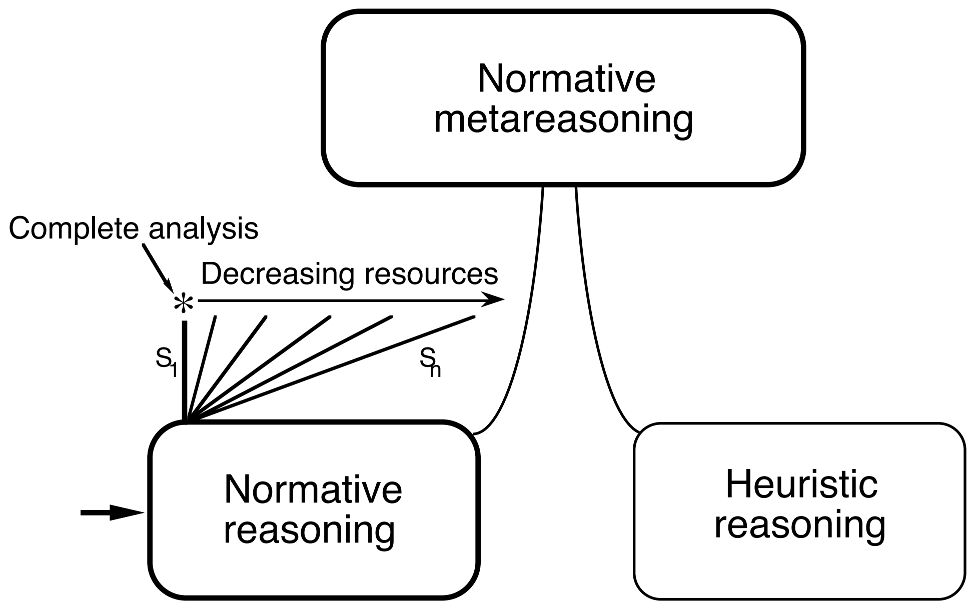


Figure 1.2: Principled control of normative reasoning.

Normative meta-analyses could be performed to choose among alternate heuristic and normative-approximation strategies; we examine the control of approximation strategies S for normative problem solving. We desire approximation procedures that can provide partial results along a spectrum of precision or accuracy as more resources are committed to problem solving. The availability of flexible strategies provides a normative metareasoner with a continuum of decisions about the amount of resources to commit to reasoning.

a single final answer: If a reasoning system has enough time for a complete analysis, a precise answer is made available. If fewer resources are available, reasoning is worthless.

We would like to have the ability to perform increasingly complete analyses as time becomes less expensive, and to trade off accuracy for timeliness as the pressure to act increases. I have pursued the development and investigation of flexible reasoning strategies that can perform analysis at different levels of detail, and that can trade off different dimensions of quality for computation time. I use *flexibility* to mean the ability of a strategy to generate results spanning a continuum of quality, paralleling the quantity of computational resource that is allocated. The ability to adjust the detail and level of normative analysis, and to appropriately focus the attention of computational resources, can help us to improve the tractability and naturalness of computation.

Figure 1.2 expresses schematically the value of *flexible* normative-approximation procedures that can provide partial results along a spectrum of precision or accuracy as more resources are committed to reasoning. The figure highlights how flexible strategies allow for the graceful degradation of the precision of the results of probabilistic inference as fewer resources are applied to the problem. Flexible strategies can be viewed as increasing the number of control decisions available to a reasoning system. Such strategies are especially valuable for a system that must reason under varying and uncertain resource constraints. We shall describe in Chapter 2 how giving a reasoner the opportunity for making decisions about the best tradeoff between time for computation and quality of results frequently translates into increased expected utility.

We can apply normative metareasoning to reason about the selection, and optimal halting time, of reasoning strategies that refine results as costly resources are expended. Designing decision-theoretic strategies with a range of precision can endow automated reasoners with the ability to respond effectively to decision-making challenges over great ranges of computational resource costs and constraints.

1.5.1 Flexible Probabilistic Inference

After exploring basic properties of flexible computation and normative control, we shall turn to the problem of computing beliefs and ideal actions under varying resource constraints. I shall present flexible approximation strategies, including *bounded conditioning* (Horvitz et al., 1989c). Bounded conditioning decomposes inference problems into sets of subproblems, and solves these subproblems in the order of their ability to refine the upper and lower bounds on a probability of interest. Figure 1.3 highlights the manner in which this algorithm refines the bounds on a probability of interest with computation. As we shall see in Chapter 5, the algorithm has been designed to perform in the spirit of the 80–20 rule; if possible, most of the inference problem is solved early on.

1.5.2 Economics of Flexible Computation

We shall investigate the ideal control of computation for the automated determination of beliefs about the truth of relevant propositions, and for determining optimal actions in the world, based on those beliefs. However, exploring the computational foundations of rationality will require a clear elucidation of general principles for controlling computation under resource constraints. Such principles can be useful for optimizing the value of computer-based problem solving in a large number of application areas.

Figure 1.4 highlights the generality of normative metareasoning about flexible computation. Beyond decision-theoretic inference, the focus of my dissertation work, we can apply normative metareasoning for optimizing the value of a variety of computational goals. In Chapter 3, we shall touch on illustrative examples that highlight the potential gains of using flexible computation and normative control of reasoning for solving tasks as diverse as sorting a file of records or searching a large tree of possibilities.

We shall explore flexibility with regard to sorting before delving into the details of flexible probabilistic inference and normative control of inference in Chapters 4 and 5. The task of sorting a file of records provides a source of pedagogical examples of flexible reasoning and of multiple dimensions of value in a partial result. We

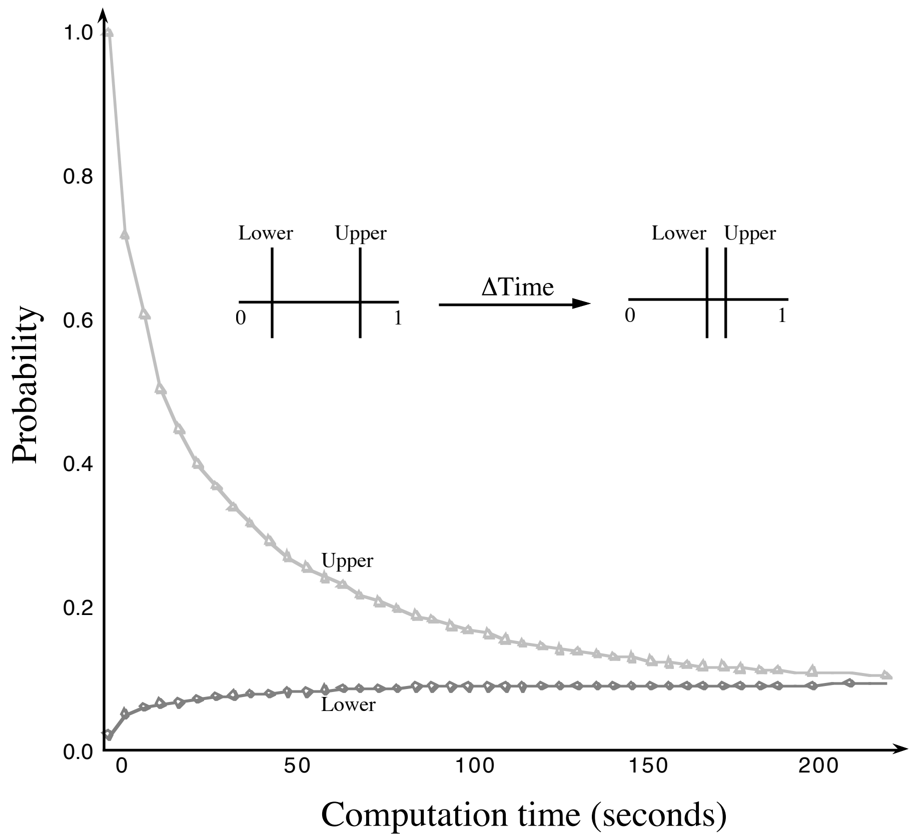


Figure 1.3: A flexible probabilistic-inference strategy.

The graph demonstrates the convergence of the upper and lower bounds on a probability of interest with the application of the bounded-conditioning approximation strategy. With computation, the interval between the lower and upper bounds is diminished.

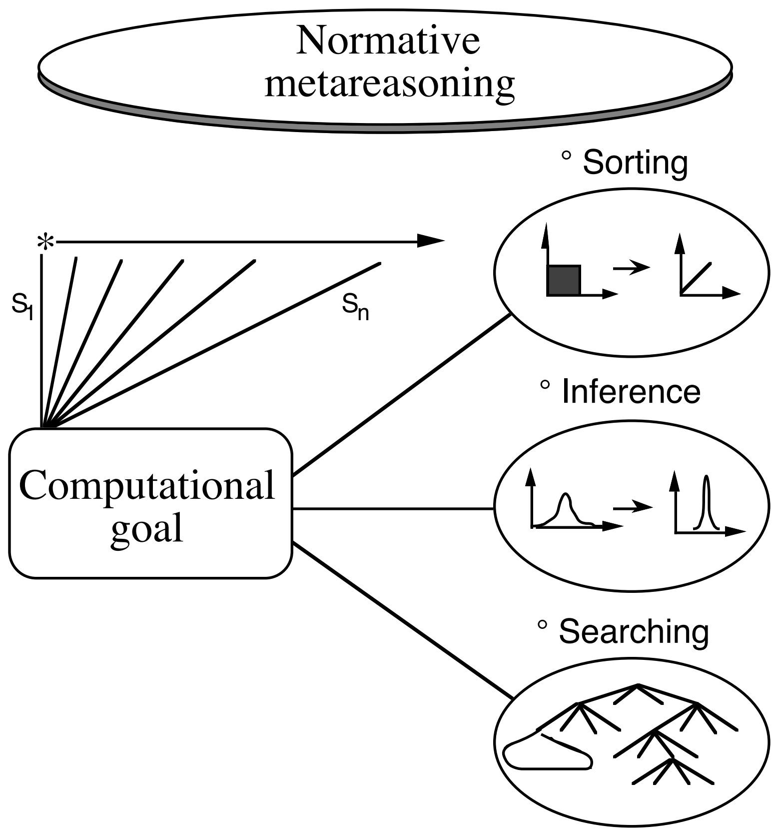


Figure 1.4: A generalization of normative metareasoning to general problem solving. We can apply normative reasoning about flexible computation for controlling the allocation of resource in a variety of computational tasks. For example, the principles of decision-theoretic control and the value of graceful degradation of complete analysis described for discussions about control of decision theory apply to sorting and searching.

shall see how several sorting algorithms can produce partial results, and can refine the results with continuing computation. I shall describe a system I constructed, named Protos/Algo, that enables us to build and inspect the value of partial sorting results, and to observe different patterns of problem refinement. The system displays partial sorts as two-dimensional graphs of points with the y coordinate equal to the sorting key of a record, and an x coordinate equal to the position of that record in a file. Protos/Algo's graphical output and associated economic analysis is displayed in Figure 1.5. These graphs show the position of record keys of different values in a file before and during the sorting process. The graph in the foreground of Figure 1.5 plots the value over time of continuing to compute.

1.6 The Protos System

In Chapter 6, I shall present the architecture and behavior of a reflective decision system, named Protos.[3] I implemented Protos to demonstrate key issues of normative metareasoning about probabilistic inference. The system is a prototype of normative metareasoners that may someday find application in expert systems, closed-loop decision-making systems, and autonomous agents that make use of large probabilistic knowledge bases for making decisions.

The operation and basic components of Protos are displayed in the schematic in Figure 1.6. Protos' architecture consists of (1) an inference base, which contains probabilistic inference strategies, (2) a probabilistic-dependency base, consisting of a belief network representing domain knowledge, and (3) a normative metareasoner. At run time, a problem-specific decision model (in the case of medical reasoning, a *patient-specific* model) is constructed from a utility model and decision problem that are passed to the system. The metareasoner uses knowledge about the efficacy of problem solving to make a decision about the best computational strategy to apply. As highlighted in Figure 1.6, the metareasoner trades off the cost of delay with the value of additional reasoning to determine the best action to take, and the time at which to take that action. In response to evidence (e.g., observations about a

[3] *Protos* is a partial acronym for *p*roject on computational *r*esources and *t*rade*o*ff*s*.

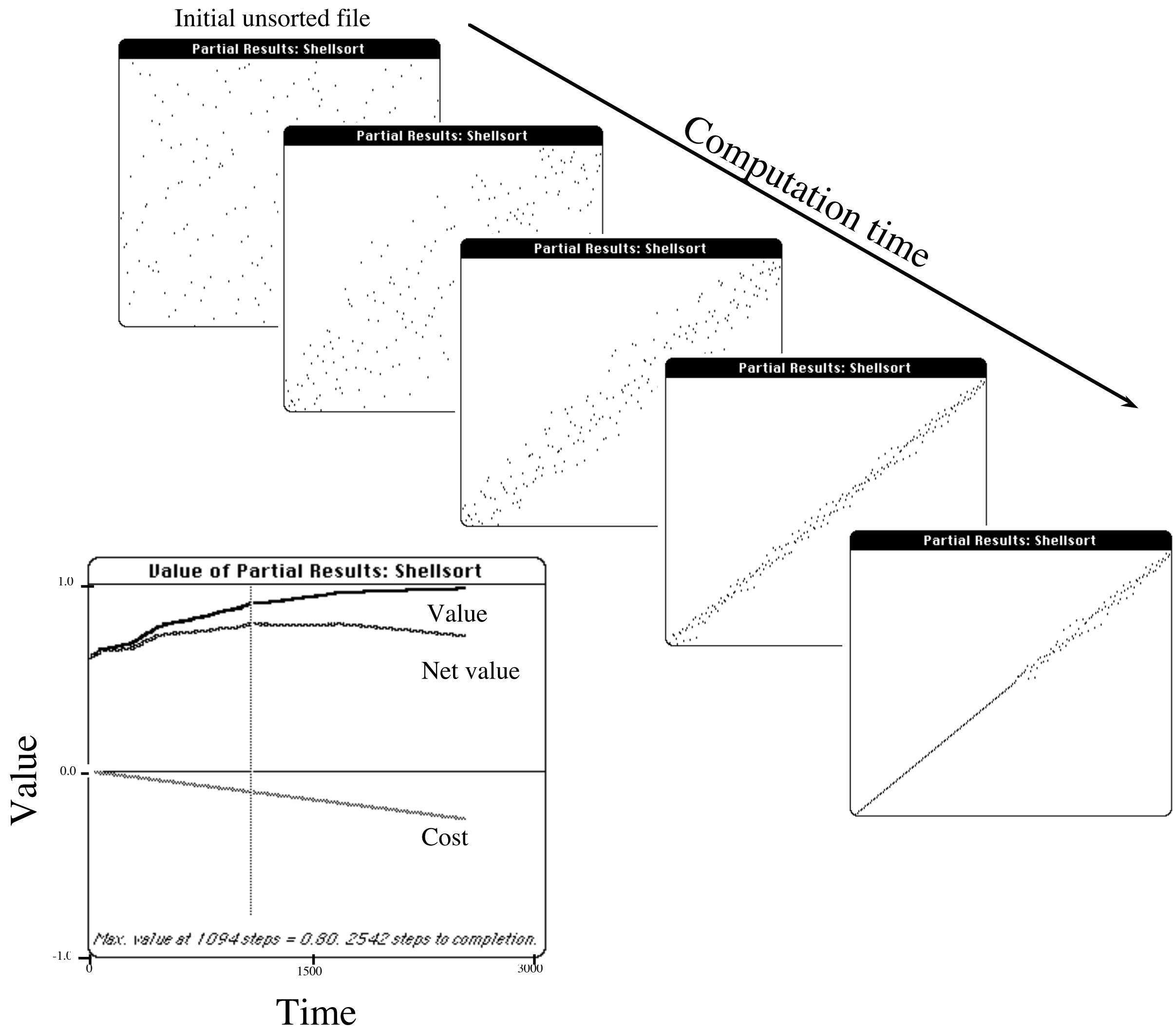


Figure 1.5: Exploration of the value of partial results.

The Protos/Algo system displays partial results and optimal halting times of different sorting algorithms. Points in each graph represent the position (x axis) and the value of the sorting key (y axis). The system enables a user to define multiattribute utility models that assign value to partial sorts. The costs and benefits of the sorting algorithm are summarized in a graph (foreground). Note that the net value begins to drop because the constant increase in computational cost begins to dominate diminishing improvements in the result.

patient's symptomology), Protos makes use of knowledge about the time-dependent utility of alternate outcomes to decide how long to dwell on an inference problem before recommending action.

As an example, let us consider the case where Protos helps a physician to decide whether a patient showing signs of respiratory distress should be given assistance with mechanical ventilation. Given information about a patient's symptoms, Protos begins to apply a flexible probabilistic inference strategy. Graphical output of Protos, used to justify the system's actions, is displayed in Figure 1.7. One graph shows the convergence with computation of the upper and lower bounds on the probability that the patient is in a state of respiratory failure, a state that requires immediate mechanical ventilation. The system also graphs changes over time in a probability threshold that dictates optimal action. The probability threshold is determined from the costs and benefits of treating a patient that may or may not have have a disease. In our example, the threshold is the probability of respiratory failure that would dictate treatment of the patient with a mechanical respirator.

Under the pressure of time, Protos can dictate action before a point probability is computed, and before the bounds on the probability that a patient has a disease passes over the threshold. Another graph in Figure 1.7 shows the expected value of continuing to compute. It is worthwhile to continue as long as the *expected value of computation* (EVC) is positive. Protos can show different perspectives on the decision making under bounded resources. Figure 1.8 is a representation of the state of belief and utility at the time a decision to act was recommended.

Protos' behavior demonstrates how ideal computation and decisions change with alterations in the time criticality of a situation. Figure 1.9 displays the same inference problem in a less critical context. In this case, a later halting time is indicated. Figure 1.10 demonstrates the partial result at the new halting time for this context. Details about the design, functionality, and performance of Protos will be presented in Chapter 6.

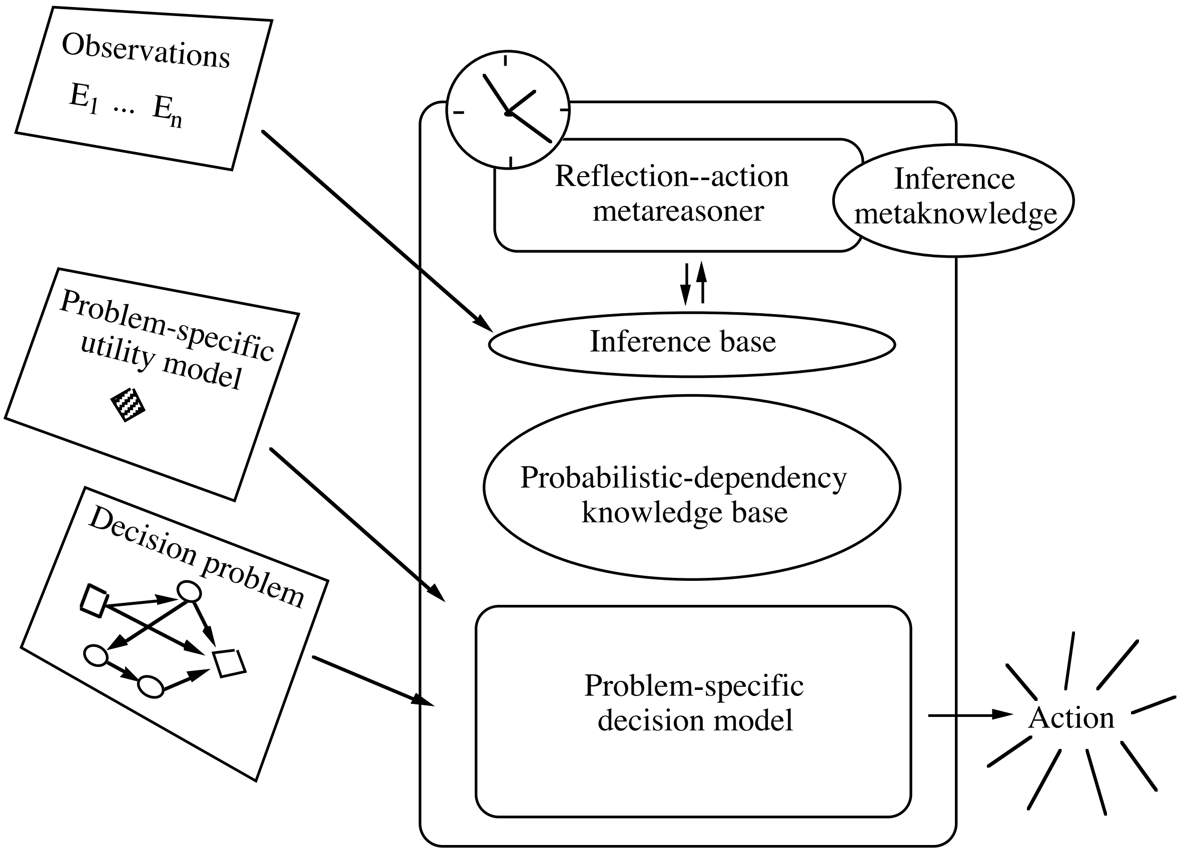


Figure 1.6: Architecture and behavior of Protos.
In response to observations (such as the symptoms of a patient in a critical-care setting), Protos decides on the most valuable reasoning strategy. Protos consists of an inference base, which makes available alternate probabilistic-inference strategies, a probabilistic dependency base, representing uncertain relationships in a problem area, and a normative metareasoner, which decides how complete an analysis should be undertaken given the costs and benefits of reasoning. This decision depends on the stakes and time-dependent utility represented in a problem-specific model that is passed to the system or constructed at run time.

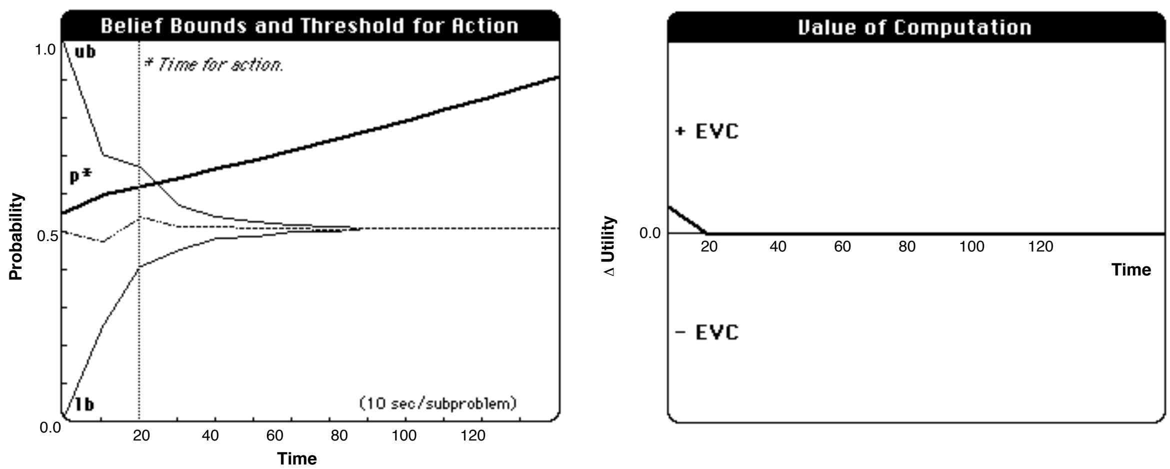


Figure 1.7: Ideal reflection before action.
Protos determines to halt inference and to recommend action after reflecting about a problem for 20 seconds. (a) A graph drawn by Protos, showing the convergence of upper and lower bounds on the probability of a crucial state (ub, lb), the mean of the current belief (line between the upper and lower bounds), and the probability threshold (p^*) for taking action. The graph also shows the ideal time for taking a treatment action (vertical line). (b) A graph of the value of continuing to compute.

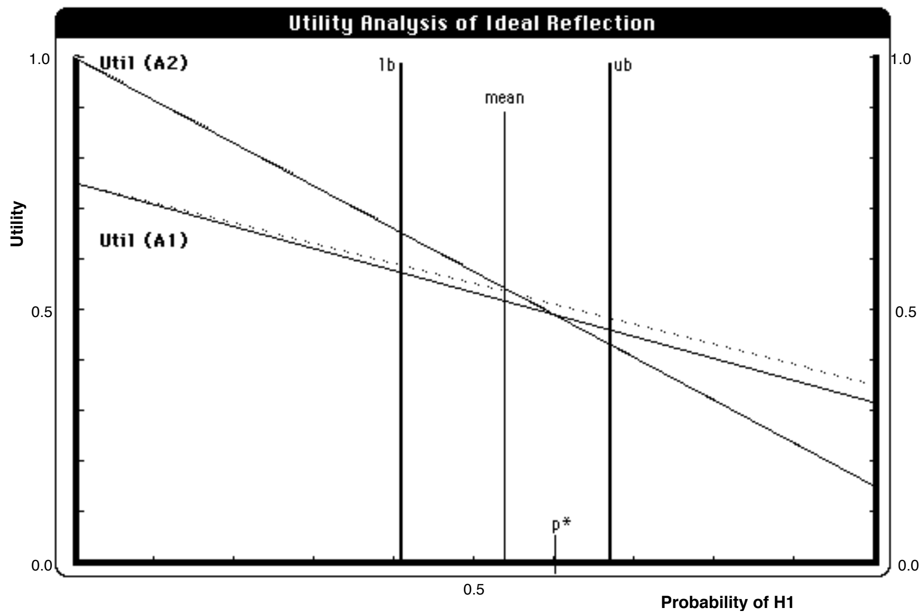


Figure 1.8: A state of partial information.

This graph represents the state of information about the decision problem from Figure 1.7 at the time Protos recommended action. The two plots are the utilities of treating (A_2) and of not treating (A_1). The lines cross at a a probability threshold (p^*) and intersect the sides of the graph at the utilities of four possible outcomes. The upper and lower bounds (lb,ub) indicate the state of the probability calculation when inference ceased. These and other components of this graph are explained in detail in Chapter 6.

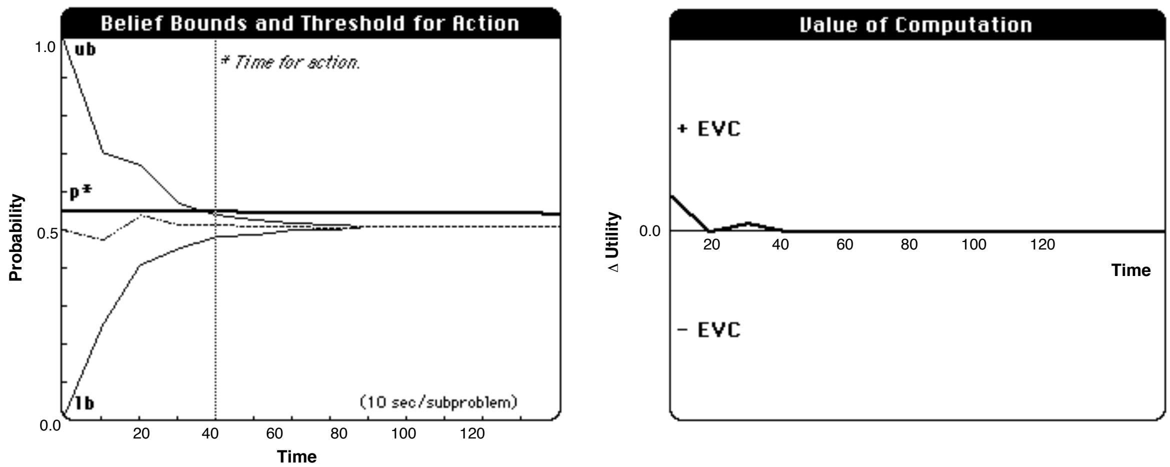


Figure 1.9: A less critical decision-making context.
Protos now reasons for 42 seconds before halting its deliberation and recommending action. (a) A graph demonstrating a new reasoning policy in a less time-critical situation. In this case, the dominance of one of the decisions is proved with greater refinement of the probability. (b) The value of computation remains positive longer in the less time-critical situation.

1.7 Dissertation Overview

I shall present basic concepts and definitions useful for the analysis of problems of computation under bounded resources in Chapter 2. In Chapter 3, I shall describe the use of multiattribute utility to probe dimensions of value of computational procedures and to reason about the costs and benefits of continuing to compute. Flexible computation and its relationship to the utility of problem solving shall be discussed in the context of sorting algorithms. We shall investigate the value of computation for decision-theoretic inference in Chapter 4. There, I shall develop closed-form solutions to the expected value of computation (EVC) for bounding algorithms. In Chapter 5, I shall introduce the flexible bounded-conditioning approximation strategy, describe its properties and limitations, and discuss possible extensions to the basic approach. In Chapter 6, I present details about the implementation of Protos. Details of the system's functionality and output shall be reviewed. Chapter 7 contains a discussion of the value of metareasoning. I also validate Protos' performance by

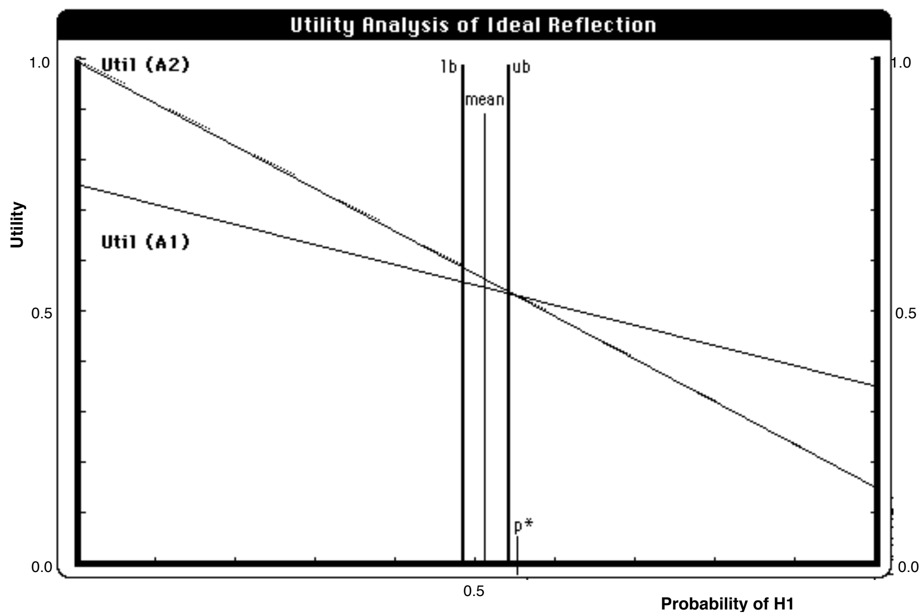


Figure 1.10: A different state of information.

In the less critical situation, Protos halts at tighter bounds on the probability of interest. In this case, the upper bound (ub) has moved below the threshold probability (p^*). When the lower bound on the probability moves below the decision threshold, the system has proved that decision A_2 is the best action.

running analyses of problems with different stakes and time criticalities. In Chapter 8, I shall present related research on normative reasoning under cognitive resource constraints. Readers interested primarily in the comprehensibility and explainability of decision-theoretic inference can review this work as a standalone chapter. Finally, I shall summarize this work and discuss future research on normative metareasoning in Chapter 9. Readers unfamiliar with decision theory, influence diagrams, or belief networks may wish to review Appendix A to understand better Chapters 4 through 6.

Chapter 2

Flexible Computation and Principled Control

To formalize reasoning and action under bounded resources, we must develop a language and a set of metareasoning procedures that allow us to consider the expected value of alternate computational methods and runtimes. In this chapter, we shall review a set of concepts and definitions about the costs and benefits of computation, and shall use these concepts to introduce the notion of partial computation and partial results. After discussing computation under uncertainty and multiple dimensions of value in partial results, we shall explore desirable properties of flexible computation for problems under varying and uncertain time constraints. Finally, we shall analyze the problem of computing the expected value of computation (EVC).

2.1 Flexibility and Control

Much of theoretical computer science has been founded on analyses of the difficulty of achieving well-defined single answers to computational problems (Aho et al., 1983;

Garey and Johnson, 1979). Investigators studying computational complexity have implicitly assigned only one of two measures of utility to computational behavior: *Either a final solution can be computed, which has maximum object-level utility, or that solution cannot be found in the time available, and the effort is worthless.* Although such an assumption has brought useful simplification to analyses of computational complexity, it has biased research toward policies that are indifferent to variation in the value of an answer, or to the costs and availability of resources. Computer scientists have done substantial work on approximation procedures that give results guaranteed to be within some measure of error from ideal solutions (Papadimitriou and Steiglitz, 1982; Lawler et al., 1985). However, little attention has been devoted to the formalization of methods that can be used to consider the costs and benefits of alternate strategies or of computing for additional quantities of time in different situations.

An examination of the dynamic nature of decision making in the real world highlights the narrow focus of worst-case time-complexity analyses. It is easy to demonstrate wide variations in the value of a result and the cost of delay to a decision maker. In situations of varying and uncertain resources, we may, in some cases, have time to solve a problem completely; in other cases, we will not have enough time. If we do not have enough time to perform the computation required to generate a complete solution, we must rely on some approximation method. Approximation strategies generate nonoptimal or *partial* results in a fraction of the time required to generate the complete or ideal answer.

We seek to increase the expected value of computation by devising machinery for custom-tailoring solution procedures to specific problems and contexts. In particular, we wish to make our computational methods sensitive to time criticality, the value of likely outcomes, and the expected refinement of solutions with additional computation. Such problem-specific information may be deterministic or uncertain. Our ability to optimize the behavior of problem-solving in varying situations hinges on the identification of one or more dimensions of flexibility that allow us to trade increases in computational resource for increases in the utility of results, or to balance alternate attributes of value in a solution.

We can introduce flexibility—and thus provide new opportunities for directing computation—at different levels of analysis. At the *strategic* level, we seek to select a reasoning strategy from a set of predefined strategies, and to determine the length of time to apply a strategy before acting in the world or reevaluating that strategy. Control decisions at the *structural level* are finer-grained decisions that determine a single or small number of computational steps. Structural control includes fine-grained decisions about the next best node to expand in a search, the best records to swap in sorting a file of records, and the best way to decompose a problem into a set of subproblems. The fundamental principles of normative metareasoning and control are insensitive to the level of analysis. For directing computation at any level of detail, we must consider (1) the expected costs and benefits of alternate computational paths, (2) the value of initiating or continuing to compute, relative to that of taking an action in the world, and (3) the costs of metareasoning and control.

2.2 Approximation and Partial Results

Before delving further into issues of control under uncertainty, let us consider the nature of approximation and partial results. We can consider an approximate or partial result to be a representation of a state of information about a complete or ideal solution. We can view a computation strategy S as generating a partial state of information $\pi(I)$ about a ideal solution $\phi(I)$ by applying a sequence of computation steps to a problem statement or *problem instance* I, and by expending some quantity of reasoning resource r—typically, computation time. We associate with each problem instance an initial state of information $\pi^o(I)$. We write

$$S[\pi^o(I), r] \rightarrow \pi(I)$$

As an example, we consider the initial positions of items in a randomly permuted file of records to be a problem instance, a sorting algorithm to be a strategy, and partial results to be the intermediate states of information representing the position of records in the file. Allocating additional quantities of time to the strategy changes the state of information about the ideal ordering of records in the file.

2.2.1 Value and Cost of Problem Solving

There are benefits and costs associated with the transformation of a problem instance into a partial or final result. We shall use *comprehensive value*, u_{c}, to refer to the utility attributed to the state of information represented by a problem instance or partial result. The comprehensive utility is a function of the *object-level utility*, u_{o}, and the *inference-related cost*, u_{i}. The object-level utility is the value associated with the information represented by the computer result and state of the world *without regard to the cost of reasoning* that may be necessary to generate the result.

Typically, we must spend time waiting for a computational result. In this case, the inference-related cost is the penalty incurred while delaying to arrive at a more accurate result. Beyond the cost of time, expensive reasoning resources may also include the memory required by a problem-solving procedure. From the perspective of a computer operating system, a cost can be associated with the dynamic allocation of memory to a specific problem-solving procedure, as this memory cannot be used by other procedures.

Assuming that the inference-related cost and the object-level utility are decomposable, and are related by addition, the comprehensive utility, at any point in the reasoning process, is the difference of the *object-level utility* of a partial result, $u_{\mathrm{o}}(\pi(I))$, and the *inference-related cost*, $u_{\mathrm{i}}(r)$,

$$u_{\mathrm{c}}(\pi(I), r) \quad = \quad u_{\mathrm{o}}(\pi(I)) - u_{\mathrm{i}}(r) \tag{2.1}$$

The net expected *change* in u_{c}, in return for an allocation of some computational resource to reasoning, is the *expected value of computation* (EVC). If we use $u_{\mathrm{o}}(\pi(I))$ and $u_{\mathrm{i}}(r)$ to refer to initial measures of the object-level and inference related costs and use $u_{\mathrm{o}}(\pi'(I))$ and $u_{\mathrm{i}}(r')$ to refer to the object-level utility and inference-related cost after expending a larger amount of resource on computation ($r' > r$), we can write

$$EVC \quad = \quad u_{\mathrm{c}}[\pi'(I), r'] - u_{\mathrm{c}}[\pi(I), r] \tag{2.2}$$

or

$$EVC \quad = \quad [u_{\mathrm{o}}(\pi'(I)) - u_{\mathrm{o}}(\pi(I))] - [u_{\mathrm{i}}(r') - u_{\mathrm{i}}(r)] \tag{2.3}$$

If the resource expenditure and the object-level value are zero at the outset of computation, we can drop the second term of Equation 2.2; thus, under this condition, the EVC is equivalent to u_c.

We shall use EVC to compare the value of alternate reasoning strategies and to reason about the length of time to apply a strategy before halting computation and acting with the information represented in the current result. We shall examine the formulation of EVC in this chapter, and shall explore tractable formulations of EVC for probabilistic-inference problems in Chapter 4.

2.2.2 Multiple Dimensions of Value in Partial Results

Most approximation methods make available some measure of error between an approximate and complete answer to a problem. Traditionally, investigators have summarized the difference between approximate and ideal results along a simple dimension of quality. We take a multiattribute utility view of approximation. For each problem instance, we associate a multidimensional *approximation space* $\mathcal{A}^I$ that contains the ideal answer $\phi(I)$ and approximations to $\phi(I)$. As we shall see in Chapter 3, dimensions of $\mathcal{A}^I$ are based on the use made of the result and are rooted in human preferences. From the perspective of an approximation space, most analyses of approximation procedures center on the ability of alternate methods to reduce a real-valued measure of distance between points constrained to a single line within the space. An example of a widely used, context-independent distance among results is the *numerical approximation*, where the distance is a measure of accuracy or precision (e.g., the result of a Taylor series carried to a particular term).

The characterization of the way an approximation strategy refines a single dimension of quality of a result can be inadequate for describing the value of alternate computational strategies because such analyses can overlook multiple dimensions of value in a partial result. Identifying multiple dimensions of quality allow us to investigate the sterotypical patterns of refinement displayed by different approximation methods. They also can help us to identify fundamental tradeoffs among attributes of quality in solving problems under specific resource constraints. Richer distance metrics $D : \mathcal{A}^I \times \mathcal{A}^I \rightarrow \mathcal{R}$ include cases where D represents the distance of $\pi(I)$ from

$\phi(I)$ along more abstract and higher-dimensional properties of a computational result. The most meaningful distance between partial results is the difference in utility itself, $u_o(\phi(I)) - u_o(\pi(I))$.

Identifying multiple dimensions of quality in a partial result can highlight directions for research on approximation strategies. For example, we can elucidate a rich multiattribute structure in partial results produced by such basic computational tasks as sorting a file of records. As an example, a librarian may ascribe value to various attributes of a partially sorted file of tardy borrowers. If a sorting task cannot be finished, he may assign value to sorting algorithms according to their respective efficiencies in identifying records on the m most tardy book borrowers out a total of n records. The value of a recommendation generated by a medical expert system in a particular context might be a function of the status of several attributes, including speed of computation, accuracy of a recommendation, and clarity of explanation. Similarly, a reasoner attempting to maximize a robot's expected utility in a complex environment will generally have to consider multiple dimensions of value in goals. For example, in making decisions about its next set of goals, a robot may have to consider the distance and accessibility of costly staples of electrical power and oil, the positions of alternative crate-stacking tasks, the speed with which it can create a new plan, and its distance from other robots that might require or lend assistance.

We can describe the value and cost of computation with vectors of object-level attributes $\vec{v} = (v_1 \ldots v_m)$, and of inference-related attributes $\vec{r} = (r_1 \ldots r_m)$. We define each scalar attribute, v_i, as a real number that lies between 0 and $v_i^*(\phi(I))$, the value of v_i in a precise answer or *final result* $\phi(I)$. From this perspective, the object-level utility of partial results is represented as a point at some distance from a desired ideal or final result $\phi(I)$ in a multidimensional space. The value of v_i reflects the degree to which a dimension of quality is embodied by a result. For all v_i, $v_i(\pi(I)) \leq v_i(\phi(I))$. Unless otherwise specified, we shall assume that the object-level utility of a result increases monotonically with increasing values of any v_i, if other attributes are held constant. To simplify my notation, I use $u_o(\pi(I))$ as shorthand for $u_o[V(\pi(I))]$, where $V(\pi(I))$ is a function that returns a vector $\vec{v}$ of relevant scalar attributes for a partial result $\pi(I)$. I shall express attributes of value when their

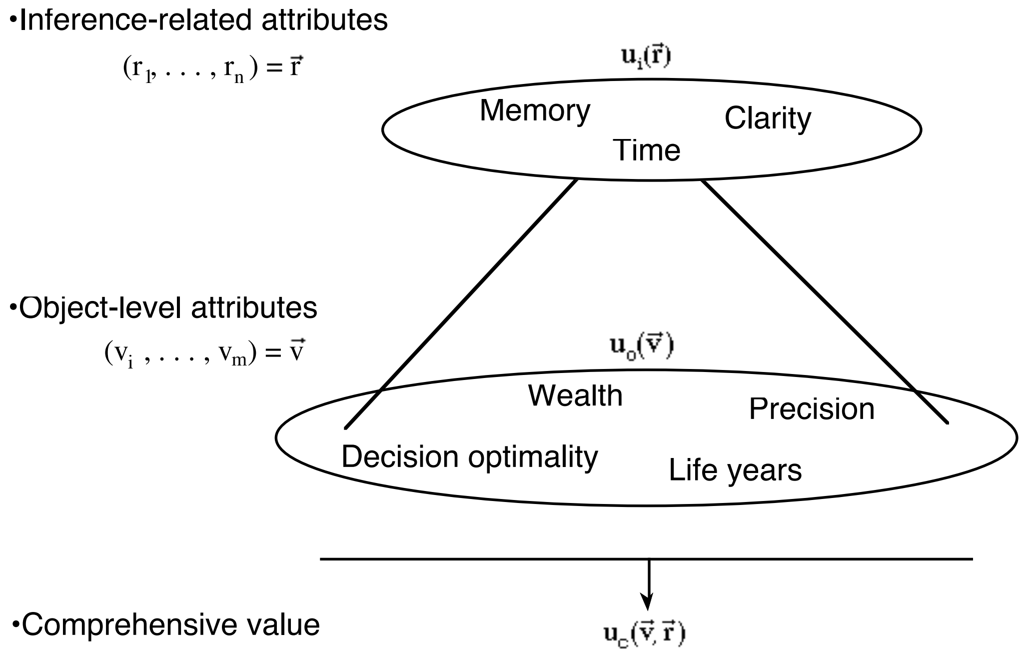


Figure 2.1: Components of value in computation.
We consider vectors of object-level and inference-related attributes in reasoning about the value of computation. Object-level attributes are dimensions of preference in a computational result considered in the context of the state of the world. Inference-related attributes are dimensions of preference that are intrinsic to the use of a reasoning system to generate an informational result. The comprehensive value of computation u_c is a function of object-level and inference-related attributes.

explicit introduction is necessary for clarity.

We shall examine the case of sorting, as a source of illustrative examples of multiple dimensions of value in partial results, in Chapter 3. We shall explore in Chapters 4 through 7 the use of approximation strategies for probabilistic inference. Such strategies produce and refine with computation different families of probability distributions that describe the uncertainty in probabilities of interest. In Chapters 4 through 9, we shall concentrate on the ideal allocation of resources for inferring recommendations from large decision models.

2.2.3 Prototypical Classes of Computation Cost

The cost of reasoning includes the cost of short-term uses of memory needed for computation and the cost of delay. We shall dwell on the cost of delay in this dissertation; nevertheless many of the key concepts can be applied to constraints in memory resource. The cost of delay can be described by several classes of penalty for delay. Functions describing $u_{\rm i}(r)$ include the *urgency* and *deadline* models (Horvitz, 1988). *Urgency* refers to the general class of inference-related utility functions that assign cost as some monotonically increasing function of delay. The *deadline* pattern refers to cases where $u_{\rm i}(r)$ is 0 until a certain amount of resource is expended. At the time of the deadline, $r = t_d$, a result must be reported, or must be used to direct action, immediately. Otherwise, the computation is worthless or a fixed cost is incurred. A deadline can be hard or *deterministic* or can be *uncertain*. An *uncertain deadline* exists in cases where a deadline is described by a probability distribution over a hard deadline.

We can construct functions to express different time dependencies by combining the urgency and deadline models. The *urgent deadline* represents situations where a cost is incurred for delay, and a deadline requires all computation to halt. *Delayed urgency* captures the case where computation is free until a particular point in time; at this time, an urgent cost model is assumed. The urgent-deadline and delayed-urgency patterns can be subdivided into certain and uncertain deadline situations. Several of the prototypical models that I have defined for penalizing delay are displayed in Figure 2.2.

2.3 Flexible Computation

We have discussed characterizing the performance of partial-result strategies with probability distributions that are conditioned on information about a problem and computation time. We have not discussed how partial results—or probability distributions that describe uncertain partial results—might change with alternate expenditures of time. Under situations of varying or uncertain resource costs and constraints, it can be valuable to refine partial results incrementally in exchange for additional

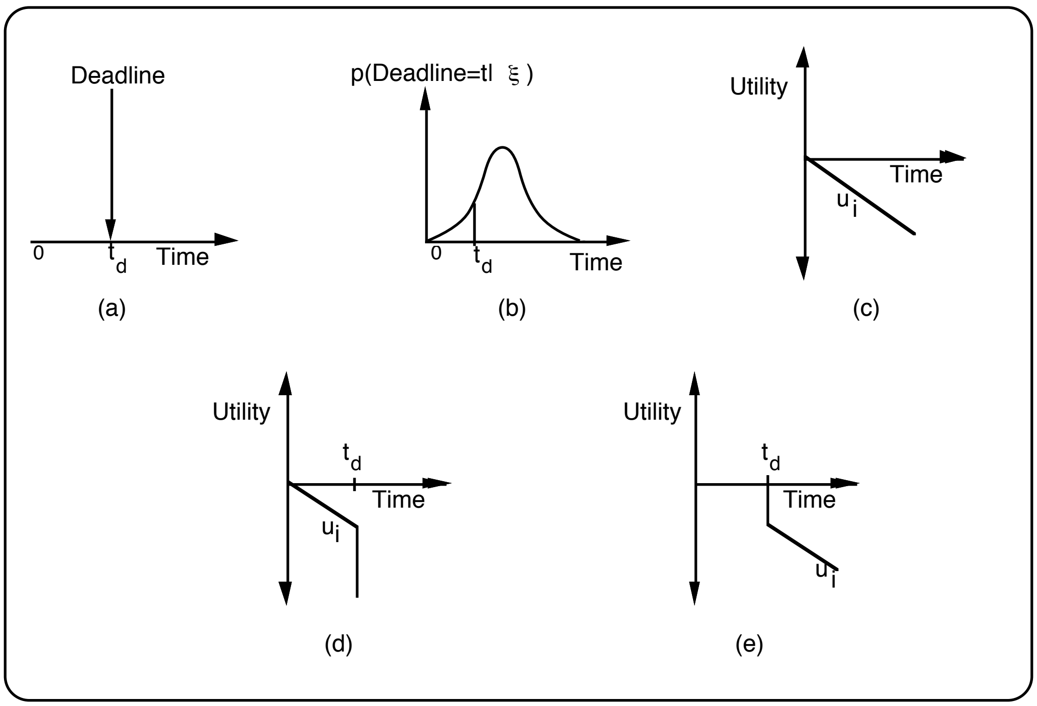


Figure 2.2: Prototypical classes of reasoning cost.

The top of the figure displays (a) deterministic deadline, (b) uncertain deadline, and the more general (c) urgency model of cost. The bottom of the figure shows two combinations of simple deadline and urgency models. In (d), the urgent-deadline model is displayed. In this situation, a cost is incurred for reasoning until a deadline. In (e), the delayed-urgency model is displayed. In this case, no cost is incurred until some future time.

quantities of computation time. Incremental refinement of partial results enables us to avoid dramatic losses given small changes in the amount of resources applied to reasoning. We often can view incremental refinement algorithms as having the ability to use the output of a previous partial analysis as the problem-instance input for additional refinement. That is,

$$S_i[\pi^o(I), r] \rightarrow \pi(I)$$
$$S_i[\pi(I), r'] \rightarrow \pi'(I)$$

Some reasoning strategies S produce results that increase in quality smoothly and monotonically with increasing amounts of computation. Other strategies produce results that do not change at all, or that become less valuable before becoming better.

2.3.1 Desiderata of Flexible Computation

We can define useful properties of computation under bounded resources in terms of the resources consumed by a reasoning strategy, and by object-level attributes of a computational result. For the definition of the properties, we shall assume that the object-level utility of a partial result is a continuous and monotonically increasing function of the value of an attribute. Desirable properties of flexible computation include *solution convergence*, *resource monotonicity*, and *resource continuity* (Horvitz, 1987c):

- **Solution convergence:** We desire our strategies to converge on the optimal object-related value at some level of resource expenditure,

  $$\lim_{r \to \infty} S_i[\pi^o(I), r] \rightarrow \phi(I)$$

  Solution convergence ensures that a flexible strategy can compute an optimal object-level result with sufficient resources. We define the *complete resources*, $r_c(S_i, \pi^o(I))$, to be the minimal quantity of resources we need to solve completely a problem instance I with that strategy. The weaker property of *partial convergence* is defined in terms of convergence on an ideal value of one or more attributes of value; that is, $v_i[\pi'(I)]$ converges on $v_i^* = v_i[\phi(I)]$ with computation.

- **Resource monotonicity:** We wish to refine attributes of a result with computation. A strategy shows resource monotonicity over a range of resource expenditure if, for any two quantities of resource r and r', $r' \geq r$, within that range, for partial results $\pi(I)$ and $\pi'(I)$ generated by computational processes $S[\pi^o(I), r] \rightarrow \pi(I)$ and $S[\pi^o(I), r'] \rightarrow \pi'(I)$,

$$v_i'[\pi'(I)] > v_i[(\pi(I)]$$

for one or more attributes v_i. We say that *strategy S shows resource monotonicity for v_i*. Because digital computation is intrinsically discrete, we typically are limited to the weaker property of *bounded monotonicity*, which constrains r and r', $r > r'$, to be multiples of a minimal quantity of resource investment $r_{\min}$, $r = nr_{\min}$, where n is a whole number.

- **Resource continuity:** We desire a strategy to show continuity in its ability to refine attributes of a result with continuing computation. For any two resource expenditures r and r' and partial results $\pi(I)$ and $\pi'(I)$ generated by computational processes $S[\pi^o(I), r] \rightarrow \pi(I)$ and $S[\pi^o(I), r'] \rightarrow \pi'(I)$, we wish, for any $\epsilon > 0$, that there exist a $\delta > 0$ such that, if $|r' - r| < \delta$, then $|v_i'[\pi'(I)] - v_i[\pi(I)]| < \epsilon$. Although we desire a continuous refinement of results with computation, the intrinsic discrete nature of digital computation introduces discontinuities. Thus, we must settle for a weaker property of *bounded discontinuity*, in which we characterize a reasoning strategy in terms of limits on the discontinuity in the refinement of an attribute with the allocation of small amounts of resources. For any two resource expenditures r and r', where $r' - r = x, x > r_{\min}$, and resulting partial results $\pi(I)$ and $\pi'(I)$, we constrain the discontinuity in the change of an attribute v_i by specifying a function $f(x)$ such that

$$|v_i'[\pi'(I)] - v_i[\pi(I)]| \leq f(x)$$

Some approximate reasoning strategies may not converge on a final answer. Other strategies may show resource monotonicity in only certain regions, or may show resource monotonicity but not converge on a final result. We use the term *spanning*

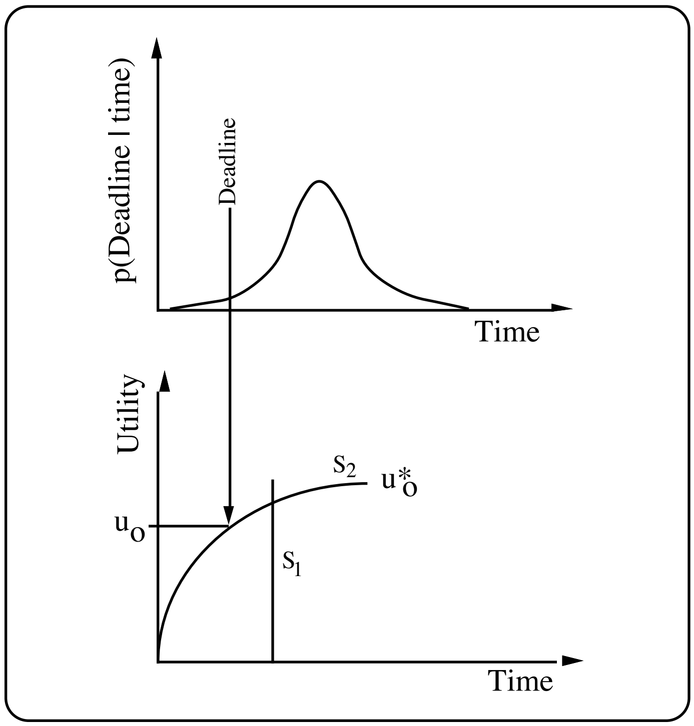


Figure 2.3: Value of flexible reasoning under varying resource constraints. The upper graph shows a probability distribution over deadline times. The lower graph shows, on the same time scale, the all-or-nothing performance of strategy S_1 and the utility of partial results generated by an incremental refinement strategy (S_2). In this case, the flexible algorithm converges on a final result with ideal object-level utility, u^*. The utility of using S_2 at any halting time is indicated by the height of the object-level utility curve at the time of the deadline (vertical line). Unlike that of the incremental strategy S_2, the utility of S_1 is worthless if the deadline occurs before the algorithm is finished.

strategies to denote reasoning methods that exhibit resource monotonicity that extends from an initial problem instance to convergence on a final result.

The desirability of solution convergence, resource monotonicity, and resource continuity (or its approximation) is founded on the pursuit of an economic "fairness" in the production of partial results. We would like to have some valuable refinement of a result in return for an expenditure, no matter how little we expend. Traditional computational methods provide us with a return only after we make a large investment. Such problem-solving "pricing schemes" are worthless when we do not have the minimum resources required to purchase a refined result.

2.3.2 Value of Flexibility Under Uncertainty

As highlighted in Figure 2.3, flexible computation is especially useful for reasoning under uncertain challenges and deadlines; flexible problem solving generates immediate object-level returns on small quantities of invested computation, and minimizes the risk of dramatic losses in situations of uncertain resource availability. The incremental nature of flexible strategies provides a reasoning system with the opportunity to select from a broad spectrum of alternate resource expenditures. Intuitions about the value of strategies that show monotonicity, continuity, and convergence can be strengthened by considering the utility of adding additional partial results to a reasoning strategy.

Consider a strategy S that generates a partial result $\pi(I)$ from an initial problem instance I with the allocation of a quantity of resource r; that is $S[\pi^o(I), r] \rightarrow \pi(I)$. Assume that S cannot generate a result with object-level utility greater than $u_{\mathrm{o}}(\pi(I))$ until a quantity of resources r' is expended, where $r' > r$. In return for r', S generates $\pi'(I)$. We call $\pi(I)$ and $\pi'(I)$ *adjacent results* for strategy S. For all-or-nothing algorithms, the initial state of information and the final answer are adjacent results. Now, let us determine the value of modifying S by adding the capability to generate results of intermediate value between two adjacent results $\pi(I)$ and $\pi'(I)$. An *intermediate result* ($\pi''(I)$ is a new partial result, generated for an intermediate allocation of resource r'', $r < r'' < r'$, such that $u_{\mathrm{o}}(\pi(I)) < u_{\mathrm{o}}(\pi''(I)) < u_{\mathrm{o}}(\pi'(I))$.

Let us consider the value of modifying an all-or-nothing strategy S by adding an ability to generate an intermediate result. We shall refer to the new strategy as S'. S requires r to generate a result $\pi(I)$. If we commit a smaller amount of resources to S, we have only the initial state, $\pi^o(I)$. S' computes an intermediate result $\pi'(I)$ for resources r', $r' < r$. We know that, if u_c increases monotonically with increases in u_o, then $u_c(\pi'(I), r'') > u_c(\pi^o(I), r'')$ for expenditures of intermediate quantities of resource, $r < r'' < r'$. To analyze how this possibility for achieving greater value increases the expected utility of a strategy under uncertainty, we consider the case of an uncertain deadline.

We shall compute the difference in the utility of applying S and S' under an uncertain deadline, described by a probability distribution $p(t_d|\xi)$ over a halting time t_d. The symbol ξ refers to the background state of knowledge that is not stated explicitly as a condition of the probability distribution (see Appendix A for a discussion of ξ). Under a pure deadline, $u_i(r) = 0$. Thus, we can compute u_c in terms of the object-level utility of partial results. We compute the expected utility by integrating over the probability density function that describes the likelihood of computing each partial result. The utility associated with S is

$$u_c = \int_{t_d<r} p(t_d|\xi) u_o(\pi^o(I)) + \int_{t_d \geq r} p(t_d|\xi) u_o(\pi(I)) \tag{2.4}$$

The revised utility, u'_c, associated with S', is

$$u'_c = \int_{t_d<r'} p(t_d|\xi) u_o(\pi^o(I)) + \int_{r' \leq t_d < r} p(t_d|\xi) u_o(\pi'(I)) + \int_{t_d \geq r} p(t_d|\xi) u_o(\pi(I)) \tag{2.5}$$

Thus, under a deadline, adding an intermediate partial result increases the expected utility by

$$\Delta u_c = \int_{r' \leq t_d < r} p(t_d|\xi) \left[u_o(\pi'(I)) - u_o(\pi^o(I))\right] \tag{2.6}$$

Endowing a reasoning system with a capability to make decisions about partial commitments of resource can be translated into increased expected utility under uncertain and varying resource constraints. We can continue to increase the utility of a strategy under an uncertain deadline by adding additional intermediate results. We refer to the preference of using strategies with greater numbers of intermediate states as *incremental dominance*.

In our analysis of the value of flexiblity, we assumed that, with the exception of the addition of an intermediate state, all else about the behavior of a strategy is unchanged. However, we do not always prefer a flexible strategy. A computer-based reasoner may have to pay a resource penalty for flexibility under uncertain and varying resource constraints: Overhead is often incurred in generating $\pi(I)$. Figure 2.3 highlights how an inflexible, all-or-nothing approach may be more efficient than a flexible approach in producing a final result and thus would be preferred in contexts where sufficient resources are guaranteed.

2.4 An Economics of Flexible Computation

Let us explore the ideal control of flexible computation strategies in the special situation where we have a functional characterization of the value and cost of computation. We rarely have such simple a priori deterministic characterizations of value and cost of computation in real-world problems. Thus, in Section 2.5, we shall resort to the use of the more general difference-equation representation to reason about changes in utility. Nevertheless, a priori deterministic characterizations can illustrate a fundamental economics of computation, and provide insight about more complex analyses of ideal computation *under uncertainty* in the quality of results that we shall review in Section 2.5 of this chapter and in Chapter 4.

The availability of functions that describe the performance and cost of reasoning strategies enables us to determine quickly the ideal object-level utility and computation time for each strategy, in turn enabling us to prove efficiently the dominance of one computational strategy over another, and to determine how long to apply that strategy. Such analyses are similar to methods employed in microeconomic theory for considering optimal levels of an industry's production of a commodity, given the value of product and expenses of manufacturing (Nicholson, 1984; Samuelson, 1973).

Let us turn to the graphical representation of the value and cost of flexible computation demonstrated in Figure 2.4. Assume that the graph represents a flexible strategy for the generation of a result needed for making a time-pressured medical-therapy decision. In this case, the object-level utility function is modeled by an

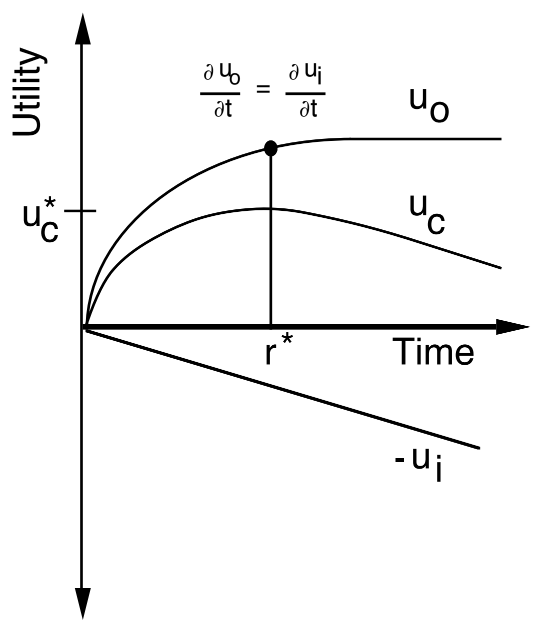


Figure 2.4: Economic relationships among components of utility in flexible reasoning. In this example, the refinement of the object-level utility (u_{o}) is modeled by a function with a positive derivative and a negative second derivative, and the inference-related cost (u_{i}) is linear with time (for clarity, we graph $-u_{\mathrm{i}}$). The optimal comprehensive value (u_{c}^*) is reached at time r^*. At this time, the rate of refinement is equal to that of the cost of delay.

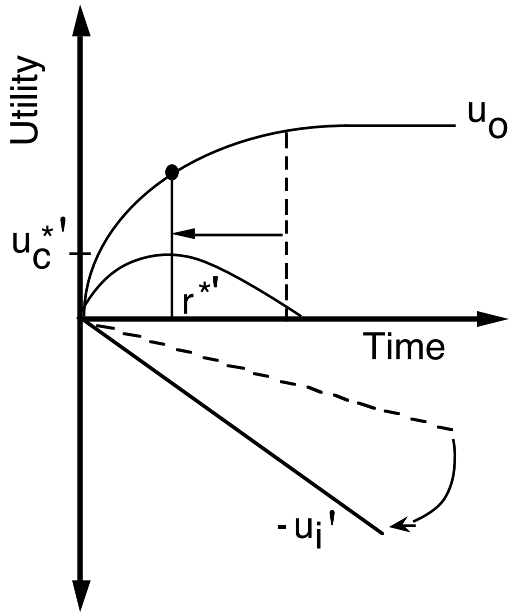


Figure 2.5: Ideal computation in a more critical context.
If we increase the cost of reasoning, from u_{i} to u_{i}', we have a new optimization. The revised optimal comprehensive value (u_{c}') is reached at a new ideal halting time $(r^{*\prime})$ when the rate of refinement object-level utility is equal to the revised rate of the cost of delay.

inverse-exponential process. Specifically, $u_{\mathrm{o}} = 1 - e^{-k(I)r}$, where k is a parameter that describes the curve as a function of an instance. The graph labeled u_{o} in Figure 2.4 displays the object-level utility delivered by a flexible strategy as a result is refined with additional time r. Assume that we find, through preference assessment, that a patient incurs a cost that grows as a linear function of time when a clinician delays an action. Figure 2.4 shows a specific linear inference-related cost function $u_{\mathrm{i}}(r) = Cr$, where C is a constant representing the time-criticality of a context. For clarity, we plot $-u_{\mathrm{i}}$. As portrayed in Figure 2.4, u_{c} rises to a maximum u_{c}^* at an ideal stopping time r^*. After we reach u_{c}^*, it is not worthwhile to expend additional resource: For each tick of the clock past r^*, we lose more in the cost of delay than we gain through additional object-level refinement.

To determine the ideal halting time and maximum comprehensive value generated by a single flexible reasoning strategy, we need to find the ideal resource allocation r^* that optimizes the u_{c}. To identify such utility maxima, we can differentiate Equation

2.1 and identify halting times where the derivative of u_{c} with respect to r is zero, and the second derivative of this function is nonpositive. At these points, the object-level value delivered per second is equal to the inference-related cost of delay, $\frac{\partial u_{\mathrm{o}}}{\partial r} = \frac{\partial u_{\mathrm{i}}}{\partial r}$. In our example, the refinement of the object-level utility of a partial result with time is modeled by a function that has a first derivative that is everywhere nonnegative and a second derivative that is everywhere negative. In this case, if the first and second derivatives of the function describing the cost of reasoning are everywhere nonnegative (as in the example), then there is a single maximum that indicates the optimal comprehensive value, u_{c}^*. In such cases, the ideal time r^* needed to compute the result with the ideal comprehensive value is reached *when the rate of refinement is equal to the cost of reasoning.* For the exponential function, we can determine from a simple optimization that $r^* = -\frac{ln\left[\frac{C}{k(S,I)}\right]}{k(I)}$, and $u_{\mathrm{c}}^* = 1 - C\left[\frac{1-ln\left[\frac{C}{k(I)}\right]}{k(I)}\right]$.

Flexible reasoning strategies allow us to optimize reasoning over large ranges of resource. As indicated in Figure 2.5, we can use such a rule to determine quickly the revised r^* and u_{c}^* with changes in the costliness of delaying action. For more complex families of value and cost functions, we may have to compare several allocations of resource to distinguish local from global maxima.

So far, we have optimized the application of a single strategy. In the case where we have a set of flexible methods, we must decide which strategy is the most valuable, in addition to what is the optimal length of time to use that strategy. In such a case, we solve for the ideal resource allocation, r_i^*, and associated global maximum, u_{c}^*, for each strategy, S_i, and then choose the strategy, S_i^*, with the greatest value. A reasoning system optimizes utility of computation by applying strategy S_i^* for $r_{S_i}^*$. Figure 2.6 shows the ideal resource allocations and object-level utilities associated with two different strategies, S_1 and S_2. Notice that the most valuable strategy to apply changes from S_1 to S_2 as the cost of reasoning increases.

2.5 Normative Metareasoning

In general, the refinement of a partial result is not a simple, monotonic function of the resources applied to reasoning. Thus, we typically do not have deterministic

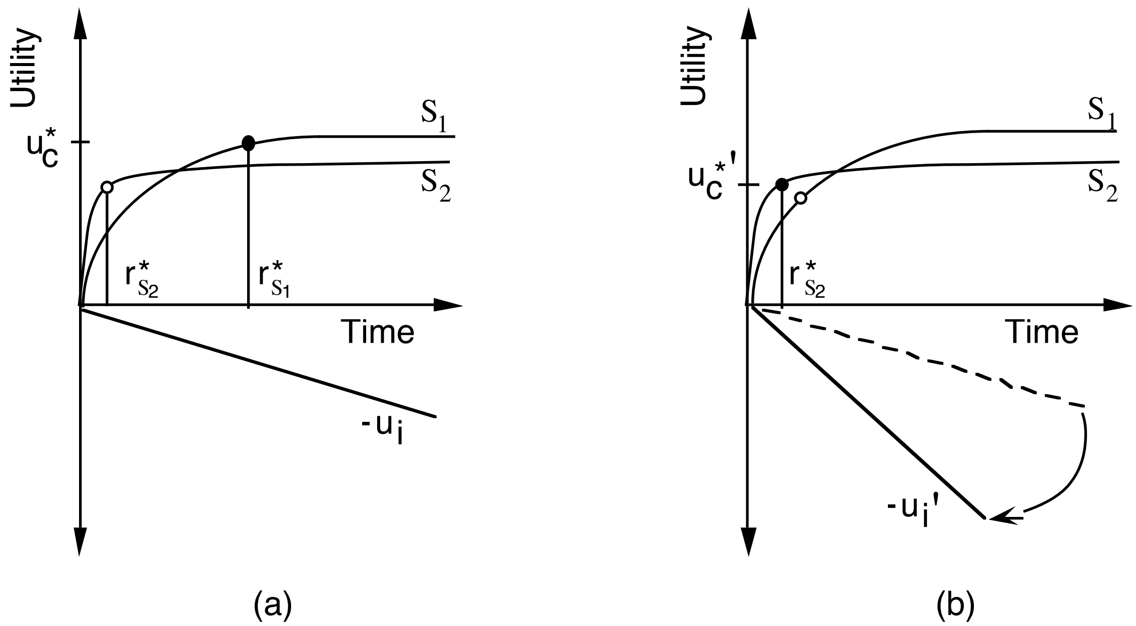


Figure 2.6: Consideration of two incremental strategies.
(a) We compare the optimal resource allocation and ideal comprehensive value of strategy S_2 (as indicated by the open circle) with strategy S_1 (as indicated by the filled circle). Given the current cost of reasoning, strategy S_1 is more valuable than S_2. (b) The strategy with the greatest ideal comprehensive value changes from S_1 (open circle) to S_2 (filled circle) for a situation of greater criticality.

information about the partial results we shall achieve with computation. We also do not have deterministic knowledge about resource cost and availability. Therefore, we cannot use simple economic arguments to decide on optimal strategies and reasoning times. In such cases, we need to reason under uncertainty to make decisions about the best means of solving a problem. To do this, we typically need to apply a fraction of available reasoning resources to deliberate about *how* to reason about a problem. Our goal is to decide efficiently which available policy has the greatest value, or whether it is better simply to take action with the current partial state. Thus, we need to calculate the EVC by summing over a probability distribution of object-level attributes expected with computation, and by diminishing the value of the object-level result by the cost of that result's generation.

2.5.1 Uncertainty in Partial Results

We are typically uncertain about the partial results that will be reached with computation. We must consider *uncertainty* about reasoning performance of a reasoning strategy, given such information about a problem instance as the size of the problem. Uncertainty about computation can be represented as a probability distribution over the partial result itself or over one or more random variables that represent the different attributes of a partial result. Under uncertainty, we may have knowledge about computational results of the form $p(\pi(I) \mid S, I, r, \xi)$, a probability distribution over different possible partial results $\pi(I)$, conditioned on the allocation of resource r, problem instance I, and strategy S. Such probability distributions that describe the behavior of a reasoning method as a function of classes of problem instances and quantities of resource can be gathered with empirical analyses, through theoretical study, or through direct assessment of an engineer experienced with the performance of an algorithm. We shall explore the acquisition of performance knowledge in Chapters 3 and 5.

2.5.2 Control of Reasoning Under Uncertainty

Let us consider a probability distribution over a single attribute of a partial result reached with the application of a unidimensional resource, r, representing the time used for computation. The extension to multiple independent attributes of result and resource requires a straightforward summation of the value associated with each dimension of value in a result. We shall assume that we have deterministic knowledge about the initial utility, $u_{\mathrm{o}}(\pi^o(I))$, of a computer-based reasoner facing a challenge, where I captures the current problem instance or *state* of the reasoner in the world. We shall focus on strategic control. I use *strategic bounded optimality* to refer to the maximization of an agent's expected utility by selecting the best strategies and lengths of time to deliberate with each strategy. A chief task in the pursuit of strategic bounded optimality is to evaluate the value of applying alternative strategies S_i to the current instance to generate a better state under uncertainty.

2.5.2.1 General Formulation

Under uncertainty, the expected value of computation is the difference between the increase in the expected object-level utility and the cost of the additional computation. Let us first consider the EVC without considering the cost of metareasoning itself.

The EVC of applying strategy S_i with a quantity of resource r is

$$\begin{aligned} \mathrm{EVC}(S_i, I, r, \xi) &= \int_{\pi(I)} u_{\mathrm{o}}(\pi(I)) \times p(\,\pi(I)\,|S_i, I,\ r, \xi\,) \\ &\quad - u_{\mathrm{o}}(\pi^o(I)) - u_{\mathrm{i}}(r) \end{aligned} \tag{2.7}$$

In complex problems, we may be also uncertain about the nature of the functions u_{o} and u_{i} used to map an object-level utility to attributes of partial results, and disutility to resource expenditures. In such cases, we can extend Equation 2.7 to sum over different weighted combination functions, in addition to considering the uncertainty over different attributes of partial results.

2.5.2.2 Rational Computation in Urgent Situations

Given several alternative strategies, a rational controller should choose the strategy S^* with the highest EVC. Under situations of urgency, we identify the strategy with the greatest EVC by optimizing Equation 2.7 for each strategy, with respect to the

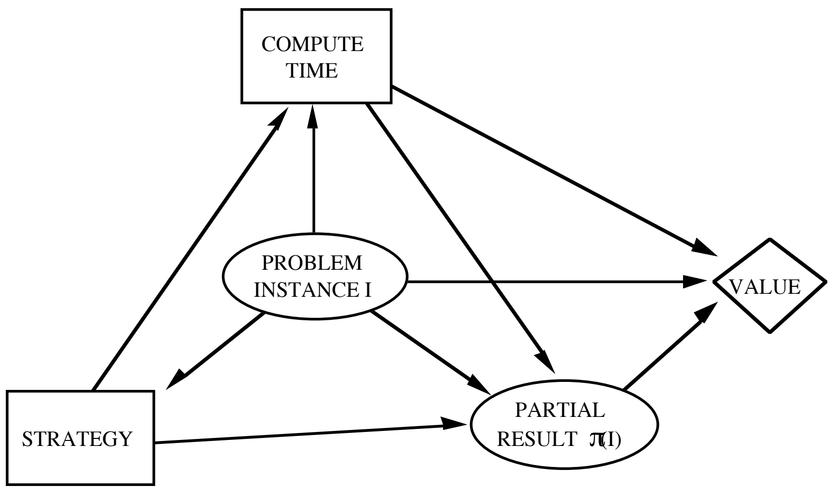


Figure 2.7: An influence diagram for normative metareasoning.
This influence diagram represents uncertainty about the performance of reasoning strategies for making strategic decisions. The goal of the meta-analysis is to identify the most valuable strategy, and to determine the length of time to apply that strategy, before acting in the world. The expected value of comptuation to a decision maker is a function of the strategy selected, the initial and final partial results, and the time allocated to the strategy. In contexts of urgency, the metalevel decision about computation time dictates with certainty the quantity of time that will be expended on the generation of a partial result.

allocated resource. We determine an ideal resource allocation r_i^*,

$$r_i^*(S_i, I, \xi) \;=\; \arg\max_r \,[\mathrm{EVC}(S_i, I, r, \xi)] \tag{2.8}$$

and examine the maximum EVC of each strategy to determine S^*,

$$S^* \;=\; \arg\max_{S_i} \,[\mathrm{EVC}(S_i, I, r_i^*, \xi)] \tag{2.9}$$

It can be useful to portray metareasoning decision problems graphically with *influence diagrams* (see Appendix A for an introduction to influence diagrams). An influence-diagram representation of the problem of determining optimal strategies and allocations of resource for urgent situations is displayed in Figure 2.7. The square nodes represent possible actions, the diamond represents the utility of alternative outcomes, and the oval nodes represent uncertain propositions. Arcs between oval nodes represent probabilistic dependencies.

2.5.2.3 Rational Computation Under a Deadline

Beyond considering the cost of reasoning, a reasoner immersed in a world of deadline situations must also wrestle with uncertainty about the amount of time available for computation. For the case of a certain deadline, we merely need to determine the strategy with the optimal EVC at the deadline time. Let us examine the more complex situation of determining the EVC under an uncertain deadline.

To determine the optimal strategy under the general urgent-deadline class of resource cost, we first calculate the optimal quantity of resource r_i^* to allocate to each strategy in a pure urgency setting, as described by Equation 2.8. Then, we consider the probability that the deadline, t_d, will occur before, versus after, that r_i^* for each strategy. Thus, the most valuable strategy S^* under an uncertain deadline is

$$\begin{aligned} S^* \;&=\; \arg\max_{S_i}[\int_{t_d<r_i^*} p(t_d|\xi)\,\mathrm{EVC}(S_i, I, t_d, \xi) \\ &\qquad + \mathrm{EVC}(S_i, I, r^*, \xi)\int_{t_d\geq r^*} p(t_d|\xi)] \end{aligned} \tag{2.10}$$

As indicated by Equation 2.10, the value associated with the use of alternative flexible and inflexible reasoning strategies depends on the probability distribution that

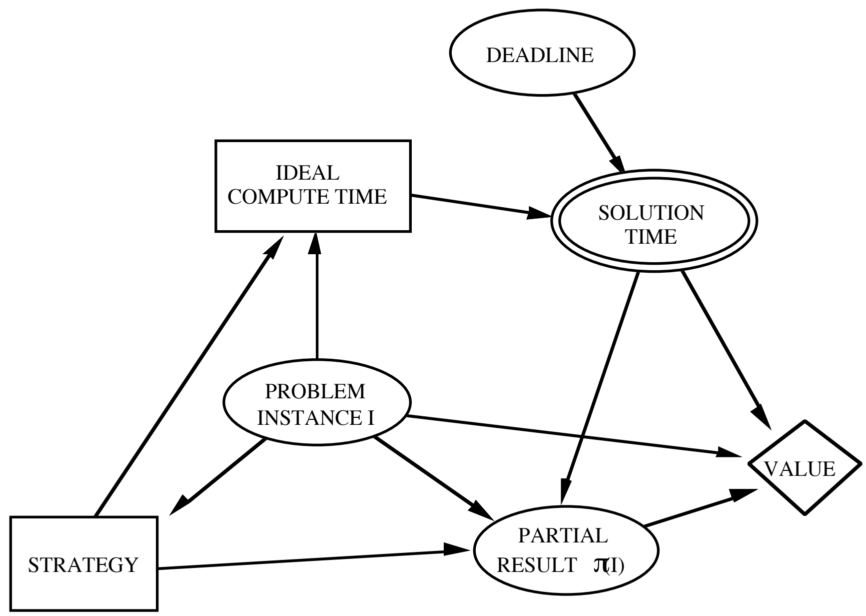


Figure 2.8: Consideration of knowledge about a deadline.

We can add knowledge about a deadline that dictates that we must cease deliberating and act with the best result generated. We introduce an explicit SOLUTION TIME node to distinguish the computation time we decide to commit to solving a problem, barring an earlier deadline (IDEAL COMPUTE TIME), from the time spent if a deadline occurs before that ideal halting time. SOLUTION TIME is a deterministic node (as indicated by the double circle) because its value is determined by the values of its predecessors (IDEAL COMPUTE TIME and DEADLINE).

describes a deadline, the uncertainty about partial results, and the cost of reasoning. An influence diagram for decisions about computation under an uncertain deadline is displayed in Figure 2.8.

2.5.3 Consideration of the Cost of Metareasoning

The influence diagram for normative metareasoning in Figure 2.7 is incomplete in that it does not represent the cost of normative metareasoning itself. Indeed, a metareasoner imposes some irrevocable cost of meta-analysis on a reasoning system. Even the relatively simple metalevel optimization for the deterministic cases, described in Section 2.4, require some resource for metareasoning procedures. Because the total computation time is the sum of solution time and metareasoning time, metareasoning diminishes the quantity of resources available for object-level computation, and, thus, reduces the quality of partial results. A central goal of research on decision-theoretic control is to identify efficient solutions and approximations to the EVC evaluation problem. We cannot always be assured that metareasoning costs will be negligible. To make explicit the inclusion of metareasoning costs, we denote by $\mathrm{EVC}^{\mathcal{M}}$ the expected value of computation, including the costs for computing the EVC. Assume that $r^{\mathcal{M}}$ is a fixed cost of metareasoning. The $\mathrm{EVC}^{\mathcal{M}}$ is

$$\begin{aligned}\mathrm{EVC}^{\mathcal{M}}(S_i, I, r) \;\; = \;\; & \int_{\pi(I)} u_{\mathrm{o}}(\pi(I)) \times p(\,\pi(I) \mid S_i, I, r - r^{\mathcal{M}}\,) \\ & -u_{\mathrm{o}}(\pi^{o}(I)) - u_{\mathrm{i}}(r) \end{aligned} \tag{2.11}$$

where $r - r^{\mathcal{M}} \geq 0$. This $\mathrm{EVC}^{\mathcal{M}}$ formula is *reflective* in that it represents the costs associated with its own calculation. The value of our metareasoning apparatus decreases as the expected difference in $u_{\mathrm{i}}(r)$ and $u_{\mathrm{i}}(r^{\mathcal{M}})$ grows. An reflective influence diagram for metareasoning, that includes a consideration of the cost of metareasoning, is displayed in Figure 2.9. In this figure, we portray metareasoning cost as being dependent on the nature of the problem instance.

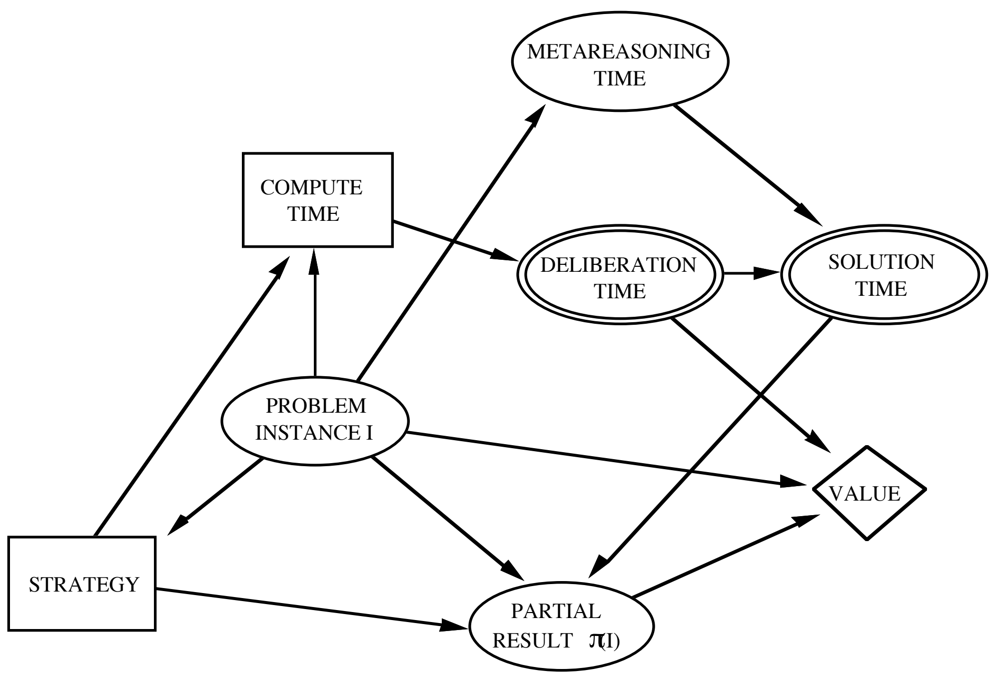


Figure 2.9: Considering the costs of metareasoning.

A normative-metareasoning system must also consider the costs of computing the value of computation. We can view the computation required by metareasoning as imposing an irrevocable tax that reduces the quantity of resources available for solving a base-level problem. We add a node representing METAREASONING TIME to the influence diagram displayed in Figure 2.7. We also add a new deterministic variable to distinguish the total time committed to problem solving (DELIBERATION TIME) from the portion of time available for solving the base problem (SOLUTION TIME).

2.5.4 Approximate and Offline EVC Analyses

I have sought to reduce the reflective control problem represented in Figure 2.9 to the problem in Figure 2.7, by elucidating tractable closed-form solutions to EVC estimation for different families of partial computation. We shall describe such tractable approximations to EVC computation for problems of belief and action in Chapter 4. These approximations are based on calculation of the value of small quantities of resource in myopic analyses, and on the use of economic analyses of functional forms to model the expected behavior of computation. Another promising EVC approximation methodology is the use of offline normative meta-analyses to generate a priori computational policies for solving problems in different problem classes. That is, we move the burden of detailed analysis to the engineering setting, and develop simple real-time control rules that react to a small set of observable problem features. In this approach, we trade off the optimization of the value of solving specific problem instances for less valuable, yet more tractable, blanket policies that can be applied to large numbers of instances. These compiled policies can provide valuable average responses to large classes of problems in situations where the complexity of meta-analysis limits the gains of finer-grained, yet more complex, real-time decision-theoretic control.

2.5.4.1 Iterative Greedy Analysis

In principle, we could apply a costly global EVC analysis to compute an ideal reasoning strategy for a decision-making agent. With such a comprehensive analysis, we begin with an initial state, and consider alternative strategy sequences and actions over large ranges of resources. As demonstrated in Section 2.4, a comprehensive analysis, for a small set of reasoning strategies, is feasible in some situations. However, global analyses typically impose unacceptable burdens on a reasoner. We can reduce the costs of EVC meta-analysis by developing approximation strategies that perform greedy optimization over a prespecified small quantity of resource R, which increases the previously expended resource r to a new total expenditure r'. In such single-step or *myopic* analyses, we continue to deliberate with additional packets of resource until

(1) a deadline is reached, (2) another strategy has a greater EVC, or (3) the expected costs of delaying for another prespecified amount of computation time R outweigh the benefits for any deliberative strategy. In the latter case, we halt deliberation and take action in the world.

A greedy formulation of EVC$^{\mathcal{M}}$, emphasizing the marginal utility of expended additional resource R, is

$$\begin{aligned}\mathrm{EVC}^{\mathcal{M}}(S_i, \pi(I), R) \;&=\; \int_{\pi'(I)} u_{\mathrm{o}}(\pi'(I)) \times p(\,\pi'(I) \mid S_i, \pi(I), R - r^{\mathcal{M}}\,) \\ &\qquad - u_{\mathrm{o}}(\pi(I)) - u_{\mathrm{i}}(R) \qquad (2.12)\end{aligned}$$

where $R = r' - r$, $\pi(I)$ is our current result, and $\pi'(I)$ is the result expected after the next computation. That is, we continue to sum over the expected future utility, weighting each possible state by the probability of achieving that outcome, and to subtract the object-level value of our current state until the expected marginal gain is nonpositive. At this point, we halt, and then act in the world. The total cost of myopic metareasoning includes the costs of metareasoning for each productive computation, in addition to a nonrefundable metareasoning penalty $u_{\mathrm{i}}(r^{\mathcal{M}})$, associated with the last meta-analysis which indicated that computation should cease.

Figure 2.10 portrays the behavior of a greedy EVC estimator that takes as inputs the current state of a partial solution, a characterization of a problem, the object-level and inference-related utility functions, and a quantity of resource that a system is considering investing in additional computation. The estimator computes efficiently the expected value of alternative strategies.

2.6 Metametareasoning and Analytic Regress

Discussions about metareasoning provoke questions about analytic regress. If control is so valuable, why not control the controller itself? And why stop there? Will our computational agents have to grapple with infinite regress to optimize their decisions? This latter question is provocative in the context of seeking a provably optimal solution. However, concerns about infinite regress have less significance given the theme of this dissertation research to increase the value of object-level analyses by invigorating base models with one or several levels of tractable meta-analyses. An assumption,

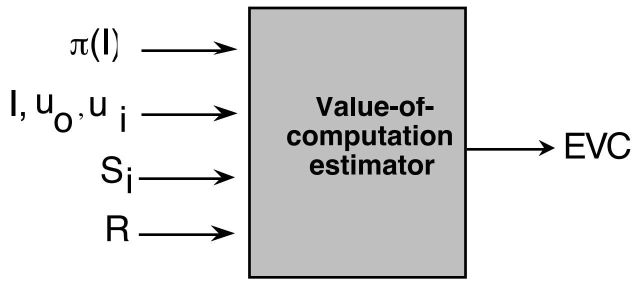


Figure 2.10: Pursuit of tractable EVC estimators.
We seek to develop tractable EVC-estimation machinery that reports the value applying of a computational strategy S_i to problem instance I for R additional seconds. Such EVC estimators make use of information about the object-level (u_{o}) and inference-related (u_{i}) components of utility, and the most recently computed partial result $\pi(I)$.

implicit in some discussions of analytic regress, is that the metareasoning problem necessarily will be at least as complex as the base problem and, therefore, will benefit from the same kind of control as the base problem. Assuming such metareasoning complexity overlooks the feasibility of enhancing the comprehensive value of reasoners by constructing simpler metalevel problems. Such metareasoners consider particular aspects of object-level problem solving, and make use of specific classes of meta-knowledge. We shall examine, in Chapter 4, tractable metalevel analyses that require a small amount of resources, relative to the time allocated to object-level problem solving. It is likely that introducing such tractable, closed-form analyses at a single level of meta-analysis offer the greatest opportunities for enhancing the value of object-level problem solving.

Nevertheless, it may be valuable to control particular metareasoning processes with metametareasoning techniques. Such control policies can be developed at design time or at computation time. Metametareasoning can be especially useful when alternate metalevel procedures are available, and where ideal metalevel deliberation may consume a significant portion of valuable computational resources. Given a problem, a context, and the details of a computational architecture, it may be best to expend at the metalevel a significant proportion of the total consumed resources. At other times, it may be best to expend few or no resources on metareasoning. Systems employing metareasoning procedures in the absence of an explicit metametalevel analysis are necessarily guided by implicit metametalevel policies. As a simple example, a system's use of a tractable metareasoner to control its problem-solving procedures relies on an implicit metametalevel decision that the benefits of metareasoning outweigh its cost. Developing metametalevel procedures to optimize the value of metareasoning is analogous to designing procedures for controlling object-level computation. In Chapter 6, I shall describe several inexpensive metametareasoning procedures that enable a reasoning system to forego more costly EVC metareasoning and, instead, to take immediate or *reflex* action. Additional investigation of the ideal partition of resources between metalevel and object-level reasoning can be found in (Horvitz and Breese, 1990) and (Breese and Horvitz, 1990). We shall return to explore problems and opportunities with analytic regress in Chapter 9.

2.7 Summary

In this chapter, we reviewed basic terminology for considering the benefits and costs in systems that must solve problems under varying and uncertain resources. I introduced the nature of partial results and discussed the assignment of measures of utility to multiple attributes of computation. After presenting desirable properties of flexible computation, we examined a simple economics of computation in settings where we have deterministic knowledge about the costs and benefits of reasoning. We then moved into the realm of characterizing uncertain performance. I introduced the problem of normative metareasoning and described a family of resource contexts in terms of urgency and deadline models. Finally, we touched on issues of metametareasoning—the control of metareasoning. In Chapter 3, we shall explore illustrative examples of the multiattribute nature of partial results with sorting algorithms. In Chapter 4, we shall apply some of the ideas discussed in this chapter to the control of probabilistic inference.

Chapter 3

Utility of Partial Results: Analysis of Sorting

In Chapter 4, we shall turn to the central topic of this dissertation: the decision-theoretic control of normative reasoning. We shall see how the ideas and formulae developed in Chapter 2 can be applied to control probabilistic inference. In this chapter, I shall illustrate key concepts of multiattribute utility in partial computation with the solution of sorting problems. I shall not attempt to address pragmatic concerns about sorting efficiency; sorting is a tractable task with a computational complexity of $O(n \log n)$. Rather, I shall use sorting as a pedagogical example of partial computation that highlights (1) multiple dimensions of value and problem-solving activity, (2) relevance of preference to the value of computation, (3) value of flexible reasoning under uncertain and varying resources, and (4) cost–benefit analyses under certainty. We can generate salient graphical examples of partially sorted files, and can inspect alternate patterns of computational activity. As we shall see, the refinement of different dimensions of partial results for sorting has significance to the solution of more difficult problems.

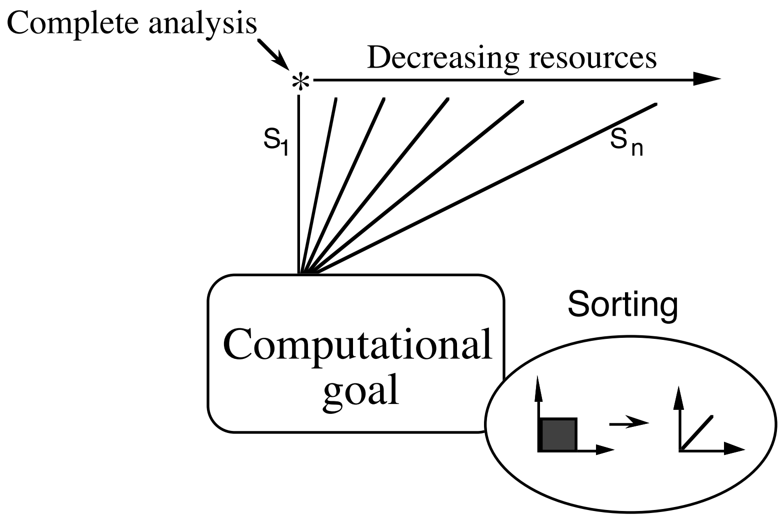


Figure 3.1: Examination of the value of an incompletely sorted file of records. We use the task of sorting a file of records under varying resource constraints as an illustrative example of partial-computation concepts.

3.1 Complete and Partial Sorting Analyses

The task of sorting a file of records has served a distinguished role as one of the most well-studied computer-science research areas (Knuth, 1973; Rivest and Knuth, 1973). Sorting is one of the central applications for computers. Investigators reported, in a 1988 study, that the sorting of files accounted for about one-fourth of all computer cycles in the world (Aggarwal and Vitter, 1988). The most efficient strategies for sorting a file of n records require $cn \log n$ time, where c is a constant factor, and n is the size of a file. Traditional versions of these algorithms—such as heapsort and quicksort—are handed a disordered file and return a completed sort. These strategies do not make partial results available. In contrast, several polynomial-time sorting strategies, such as shellsort, selectionsort, and bubblesort continuously refine one or more object-level attributes of a partial sort. On average, shellsort requires approximately $cn^{1.5}$ to sort a file completely. Selectionsort and bubblesort have quadratic run times. Although the average-case and worst-case completion times of many sorting strategies have been characterized, the exact time required by an algorithm to sort a specific file depends on the details of the order of records in a file.

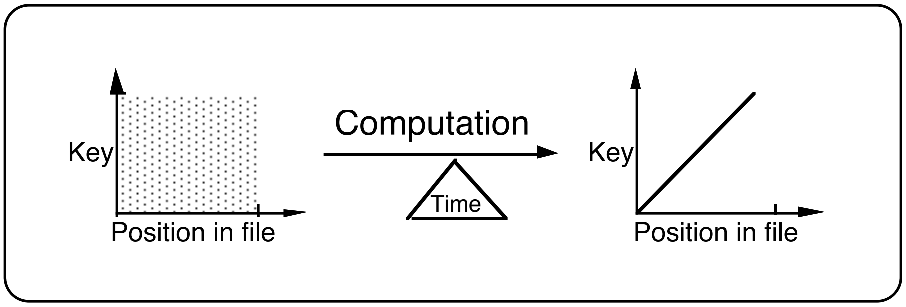


Figure 3.2: A representation of partial-sort results.
We can represent partial sorts as a two-dimensional graphs of points. This schematic displays a representation of an initial sorting problem-instance (left) and a completely sorted file (right). Sorting algorithms, fueled by time and memory, convert a disheveled file into an ordered list, with each item in its correct position.

In contrast to the long history of investigation on the quantity of resource needed to complete a sort (Knuth, 1973), there has been little work on the analysis of the ability of different sorting algorithms to produce valuable approximate results.

3.1.1 Protos/Algo: A Tool for Exploring Partial Results

I constructed a program, named Protos/Algo, for exploring the partial results of alternative sorting strategies under uncertain and varying resources. An investigator can use Protos/Algo to generate randomized files, and to examine the sorting process exhibited by different sorting algorithms. Protos/Algo has a facility for allowing a user to specify a multiattribute preference model that summarizes her preferences about partial results (Horvitz, 1988). The system can be instructed to display the position and value of records in a file graphically, and to graph the value of partial results, based on the user-specified utility model as the process of sorting progresses. The system can also perform economic analyses of the costs and benefits of computation, and graph u_{o}, u_{i}, and u_{c} as a partial result is refined.

Protos/Algo displays partial sorts as two-dimensional graphs of points, as portrayed in the schematic in Figure 3.2. The y coordinate of the graphs denotes the value of the sorting key of a record and the x coordinate denotes the position of that record in a file. With this representation, a completely randomized file appears as a

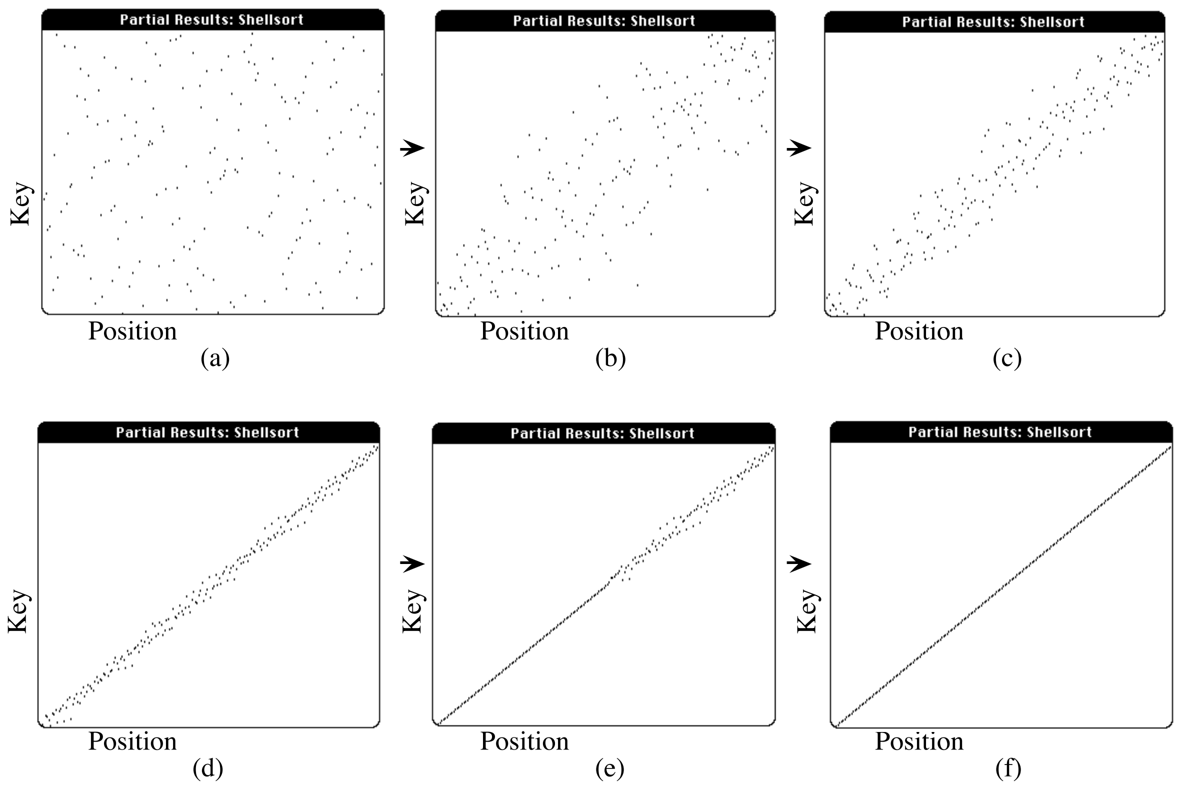


Figure 3.3: A Protos/Algo display of a stream of results produced by shellsort. With increasing amounts of resource, a problem-instance of a randomized file (a) is transformed [graphs (b) through (e)] into a final result (f).

cloud of points. A completely sorted file appears as a diagonal curve. For problem instances consisting of a randomly mixed file of n records, containing exclusive keys of values $1, \ldots, n$, the final result is a straight line, indicating that records of sort key value x are in position x.

Figure 3.2 represents the all-or-nothing conception of sorting. In contrast to this traditional view, Protos/Algo's output can reveal a stream of intermediate results, and, thus, the pattern of activity produced by sorting algorithms. The sequence of partial results generated over time by shellsort is exhibited in Figure 3.3. Figure 3.4 displays Protos/Algo output describing the behaviors of shellsort, selectionsort, and bubblesort algorithms simultaneously.

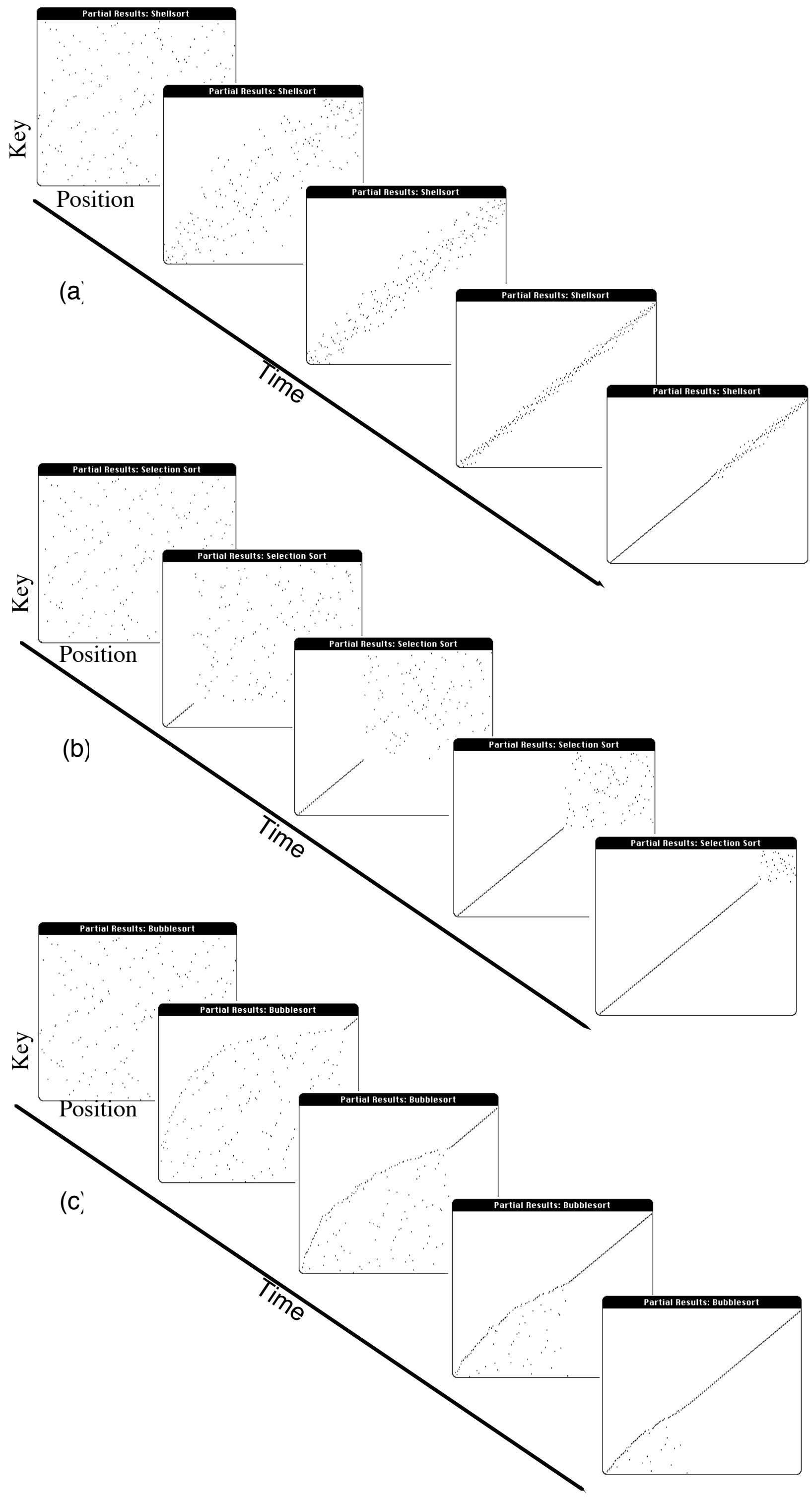


Figure 3.4: Patterns of refinement of three sorting algorithms.
The activities of (a) shellsort, (b) selectionsort, and (c) bubblesort are captured by these sequences of partial results. In each case, the sorting strategy is handed a randomized file and, with sufficient computation, transforms the result into a completely sorted file.

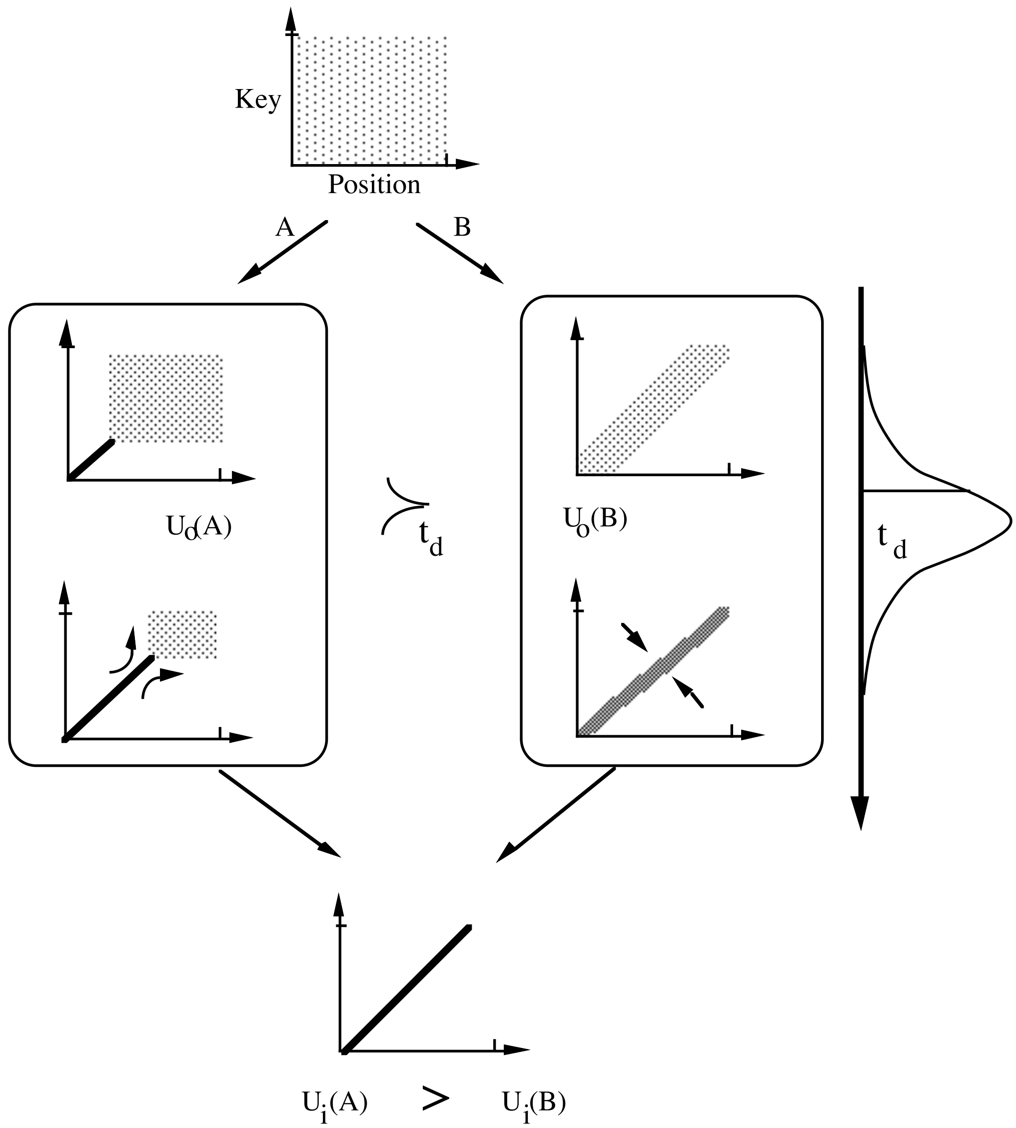


Figure 3.5: Considering the relative abilities of two algorithms under a deadline. We wish to consider the value of different problem-solving methods under uncertain resources. The probability distribution over the deadline, t_d, at the left represents uncertainty in the time the sorting algorithm must halt.

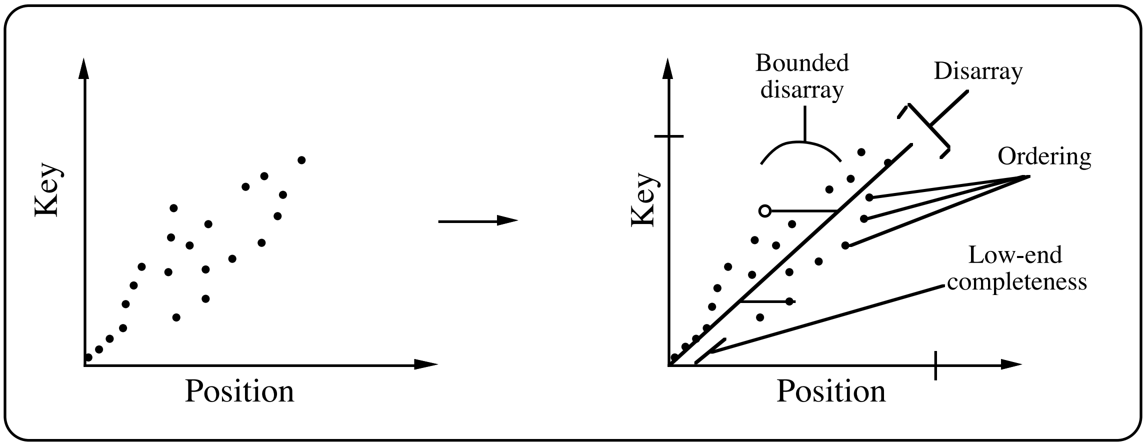


Figure 3.6: Inspection of the utility structure of a partial sort.
By focusing on users' preferences, we can reason about the relative value of different partial results. This figure shows us several components of a partial sort. The record represented by the open circle is at the maximum distance from its final location and so represents the partial result's *bounded disarray*.

3.2 Attributes of Value in a Partial Sort

Let us assume that we may not have enough time to complete a sort. Which algorithm should we choose? How long should we run that algorithm before stopping and using our result in the world? These questions provoked me to examine the value of different *partial sorts*. Figure 3.5 schematically represents the problem of choosing among two computational policies under an uncertain-deadline situation.

We can refine the definition of desired attributes by assessing the preferences of users. As an example, given a choice of partial-sort results, a university librarian might prefer a partial sort that would allow him to identify and order the largest possible group m of most tardy book borrowers out of n offenders. Another librarian might not require a precise order over identified offenders. She might prefer instead to use a sorting algorithm that can identify in the same time a larger group of g offenders that have books that are overdue by w weeks. As highlighted in Figure 3.6, a close inspection of a partial sort can reveal basic dimensions of quality.

We can define a set of sorting attributes and explore the trajectories of the partial results generated by several sorting strategies. Dimensions of value that may be useful in characterizing a partial sort include the following:

- *Partial order*: the proportion of the file in which records at positions of increasing value have keys of increasing values
- *Disarray*: the average absolute distance between the current locations and expected final locations for records in a file or within specified portions of a file
- *Bounded disarray*: an upper bound on the distance between the current position and final position for any record in a file.
- *Low- and high-end completion*: the contiguous length of a file, starting from the low or high end of the file, that contains records that are currently in the positions they will occupy after the file has been completely sorted.

Each of these attributes of a file, v_i, is defined to range between 0 and 1, where 1 is the value of the attribute in a completely sorted file, $v_i(\phi(I))$.

Starting with an initial file consisting of randomly permuted records, alternative algorithms demonstrate stereotypical patterns of refinement along the different dimensions. For example, shellsort is striking in its ability to refine disarray and bounded disarray gracefully; selectionsort is efficient for refining low-end completion. Figure 3.7 displays the manner in which shellsort enhances the disarray, the low-end completion, and the partial order of a randomized file of 100 randomized records. Figure 3.8 shows the comparative refinement of partial order, low-end completeness, and disarray for shellsort, selectionsort and bubblesort for files of size 100. The shapes of these curves are invariant to the size of the problem instance at hand.

Alternative trajectories through a multiattribute value space are portrayed schematically in Figure 3.9. In the figure, the x and y coordinates represent two attributes, $v_i(\pi(I))$ and $v_j(\pi(I))$, of a sort (e.g., completeness and partial order). The z coordinate represents the time required to generate a particular result. The figure depicts the alternate trajectories that two algorithms take through a multiattribute space, as defined by a sequence of points with coordinates (v_i, v_j, t), representing partial

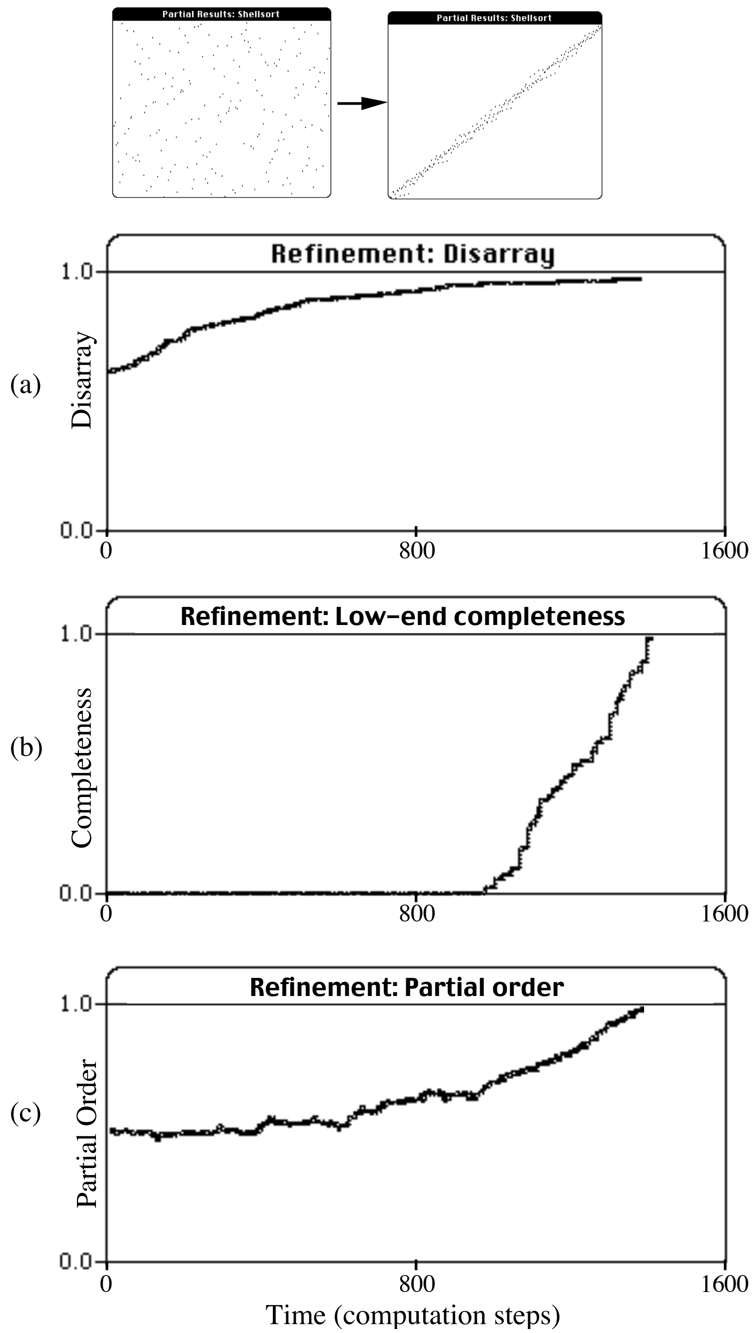


Figure 3.7: The refinement of a partial sort by shellsort.

These graphs display the ability of shellsort to refine three attributes of value in a partial sort: (a) disarray, (b) low-end completeness, and (c) partial order.

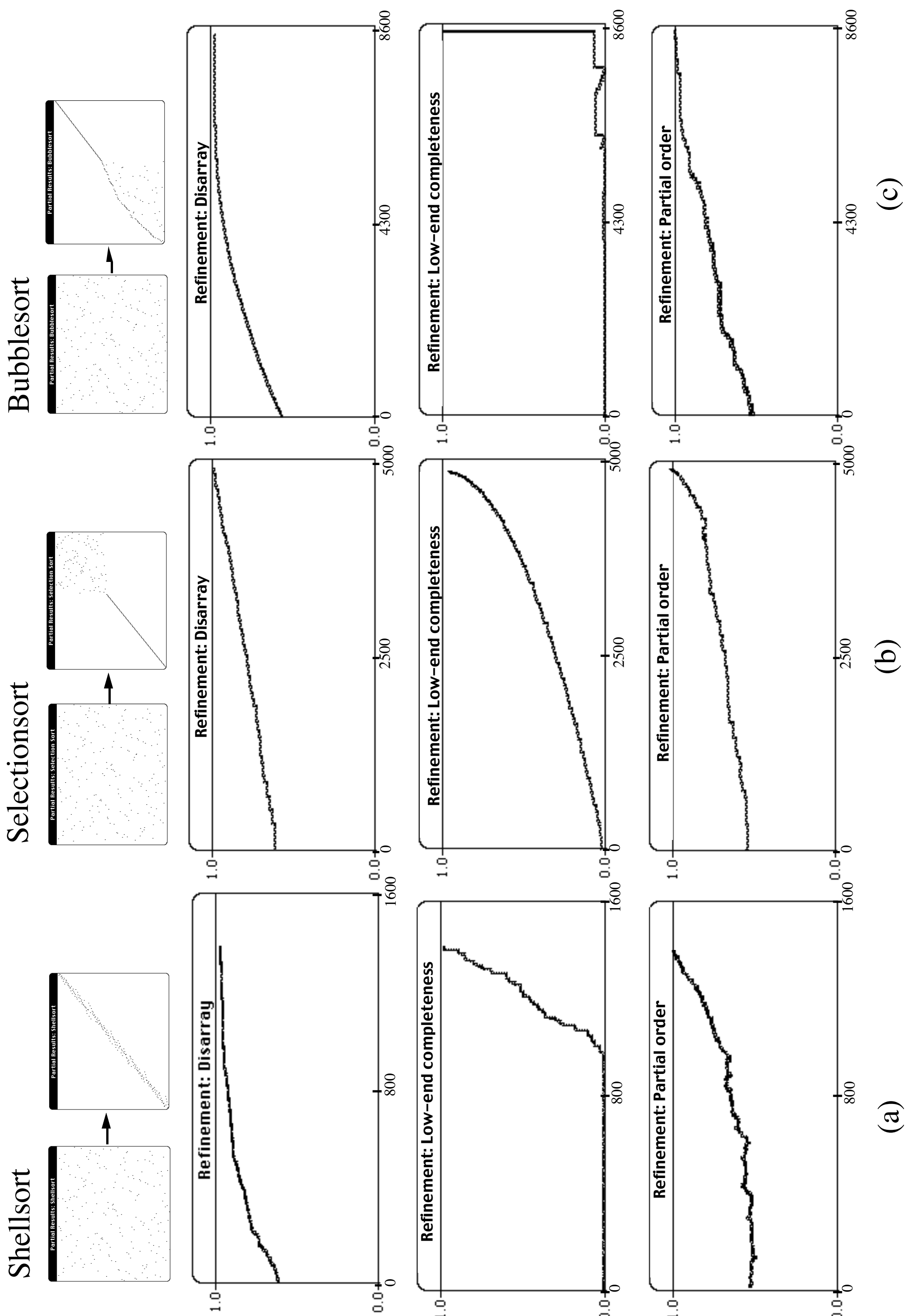


Figure 3.8: Refinement by three algorithms.
These graphs display the abilities of three algorithms to refine the same attributes of value in sorting a file of 100 records: (a) shellsort, (b) selectionsort, and (c) bubblesort.

results generated over time t. The trajectories terminate on reaching a final result, represented by a vertical line at point (v_i^*, v_j^*) in the x–y plane.

3.2.1 Combination of Multiple Attributes

We can employ preference models for computation that assign utility to multiple attributes of partial results. Parametric utility models can be personalized through the assessment of constants that specify how different attributes are weighted by an individual. The value assigned to alternative results is deemed to be a function of a set of attributes or an n-tuple, That is,

$$u_o(\pi(I)) = f(a_1, \ldots, a_n)$$

As an example, we may be able to express preferences about sorting with a parametric multilinear value equation

$$u_o(\pi(I)) = i(a_1) + j(a_2) + k(a_3) + \ldots + m(a_n)$$

where i, j, k, and m are weighting constants that can be changed to generate valuation functions that approximate the preferences of different classes of user. As this model assumes value to be a linear function of attributes, an individual subscribing to this utility model would behave as though attributes i, j, k, and m are independent from one another. If the sorting example were to be extended, genuine utility models might be assessed and validated through utility-assessment techniques (Howard, 1970; Raiffa, 1968).

Figure 3.10 demonstrates how the value of a partial result produced by shellsort depends on the details of the preference model. In this case, we manipulate the relative weighting of a two-attribute preference model, based on low-end completeness and disarray. The graph in Figure 3.10(a) shows the value of the sort for a preference model weighting low-end completeness by 0.7 and disarray by 0.3. The graph in Figure 3.10(b) displays the value of the partial results for a preference model weighting low-end completeness by 0.2 and disarray by 0.8.

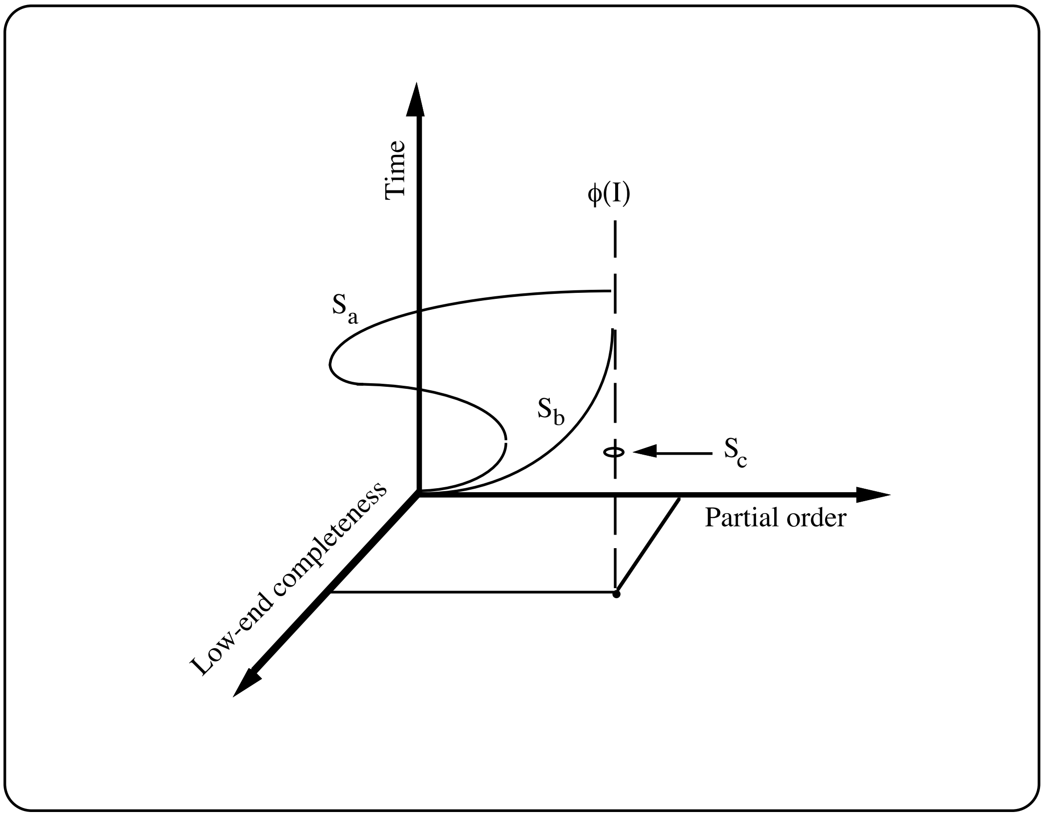


Figure 3.9: Trajectories through a multiattribute space.
Different algorithms take different paths through a multiattribute space representing dimensions of value in partial results. This three-dimensional schematic represents the paths of two incremental sorting strategies (S_a and S_b) as the amount of time allocated increases. The strategies halt when they reach a final result $\phi(I)$, represented by a line piercing a plane defined by two attributes. A third all-or-nothing strategy S_c, is shown to generate a complete sort at a time indicated by the height of the circle.

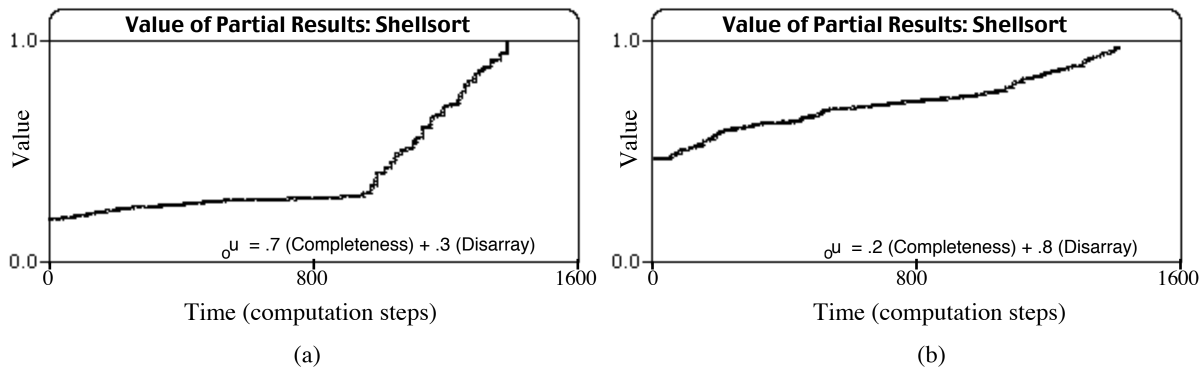


Figure 3.10: Value of shellsort for two preference models.

These graphs show the value of two different preference models for the same results over time: (a) the value of the sort for a preference model weighting low-end completeness by 0.7 and disarray by 0.3, and (b) the value of partial results for a preference model weighting low-end completeness by 0.2 and disarray by 0.8.

3.3 Economic Analyses of Sorting

Although there is often clear dominance of the best sorting algorithm to select given a file, based in large run-time differences, a variety of experiments with randomly permuted instances has demonstrated ranges of file sizes where the best algorithm to apply is sensitive to the availability and cost of resources, to the nature of the object-level and risk preferences of an agent, and to the structural details describing the refinement of results by strategies (Horvitz, 1988).

3.3.1 Sensitivity to Preferences and Resources

Decisions about a preferred sorting algorithm can be sensitive to details of the preference model function and resource constraints. Figure 3.11 shows a comparative analysis of shellsort and selectionsort for two different preference models. Figure 3.11(a) shows the value of the two sorting strategies for a preference model that weighs low-end completeness by 0.7 and disarray by 0.3. With this preference model, the two strategies perform equivalently in the early portion of the analysis, and shellsort dominates after 400 computation steps. However, if we modify the preference

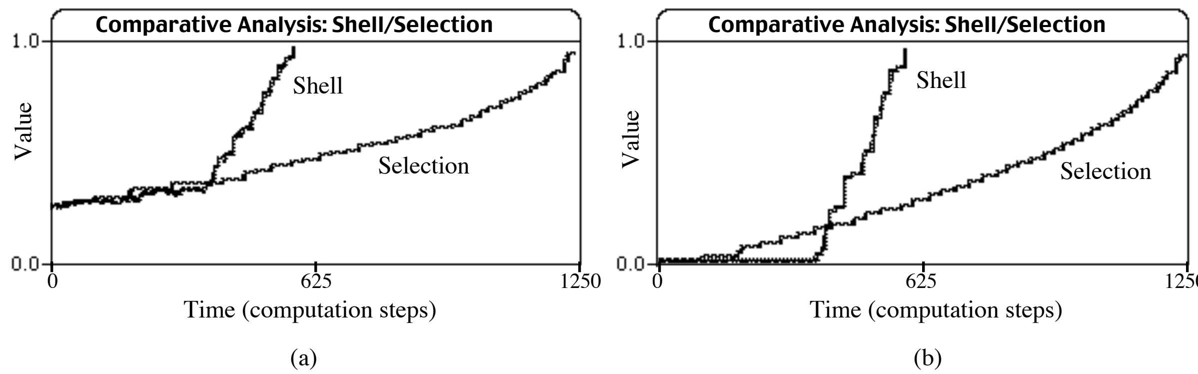


Figure 3.11: Sensitivity to preferences under scarce resources.
Under a deadline situation, shellsort can be dominated by the slower selectionsort. (a) Given a preference model of 0.7 low-end completeness and 0.3 disarray, shellsort dominates selectionsort. (b) However, given a preference model of 0.5 low-end completeness and 0.5, disarray, selectionsort dominates shellsort for a length of time.

model and weight both attributes equally, we find a length of computation time where selectionsort is more valuable than shellsort. Under a deadline situation in that period of time, selectionsort would be preferred to shellsort, even though shellsort dominates in the later phases of the sorting, and finishes the problem earlier. The values of shellsort and selectionsort assigned by this preference model is displayed in Figure 3.11(b).

In Figure 3.12, we move beyond a comparison of polynomial algorithms, and view the performance of the all-or-nothing heapsort algorithm. Although the $O(n \log n)$ heapsort is faster than shellsort and selectionsort, if a deadline occurs at some time before completion $\phi(I)$, $u_{\mathrm{o}}(\pi(I)) = 0$. In fact, the inference-related delay can incur a net cost. Thus, under resource constraints, we can generate a more valuable result by committing to the more conservative, yet more graceful $O(n^{1.5})$ shellsort.

3.3.2 Balancing Costs and Benefits of Computation

Application of the normative metareasoning methods described in Section 2.5 to the case of sorting requires that we assess probability distributions by sampling the refinement of important attributes of value for different amounts of run time. As indicated

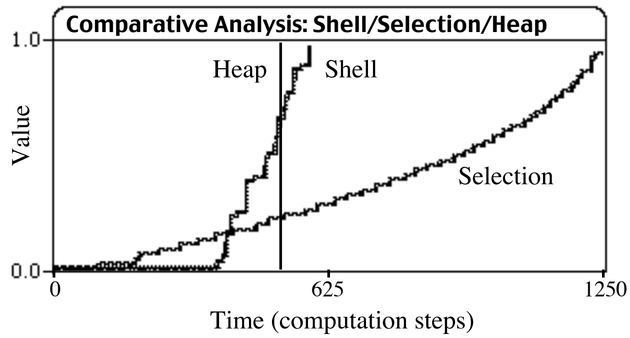


Figure 3.12: Flexible versus all-or-nothing approaches.
We add a consideration of heapsort. In situations where great cost or a deadline is expected before the completion of heapsort, we may wish to derive partial value by committing to an analysis with shellsort.

by Equation 2.8, these probability distributions are conditioned on attributes of the problem instance that are useful for distinguishing among problem instances. Such attributes include the size and initial disarray of the file. Approximate approaches to this problem center on modelling the ability of an algorithm to refine different dimensions of value with parameterized functions, and performing economic analyses, as described in Section 2.4.

Protos/Algo can model situations of different urgencies. The system can be used to identify the strategy in its library of sorting algorithms that is most valuable for a problem and urgency, and to decide on an optimal allocation of resource for that strategy. Figure 3.13 shows an analysis of the optimal halting time for shellsort, using a preference model based on the attribute *disarray.* The graphs indicate how a change in urgency can indicate that less sorting should be performed.

3.4 More Sophisticated Control

The graphs produced by Protos/Algo can help us to visualize the economics of alternative strategies and to appreciate the multiattribute nature of partial results for sorting. Although we reviewed the benefits and costs of sorting under time pressure for its visual examples of multiattribute utility, it is possible, nevertheless, to develop

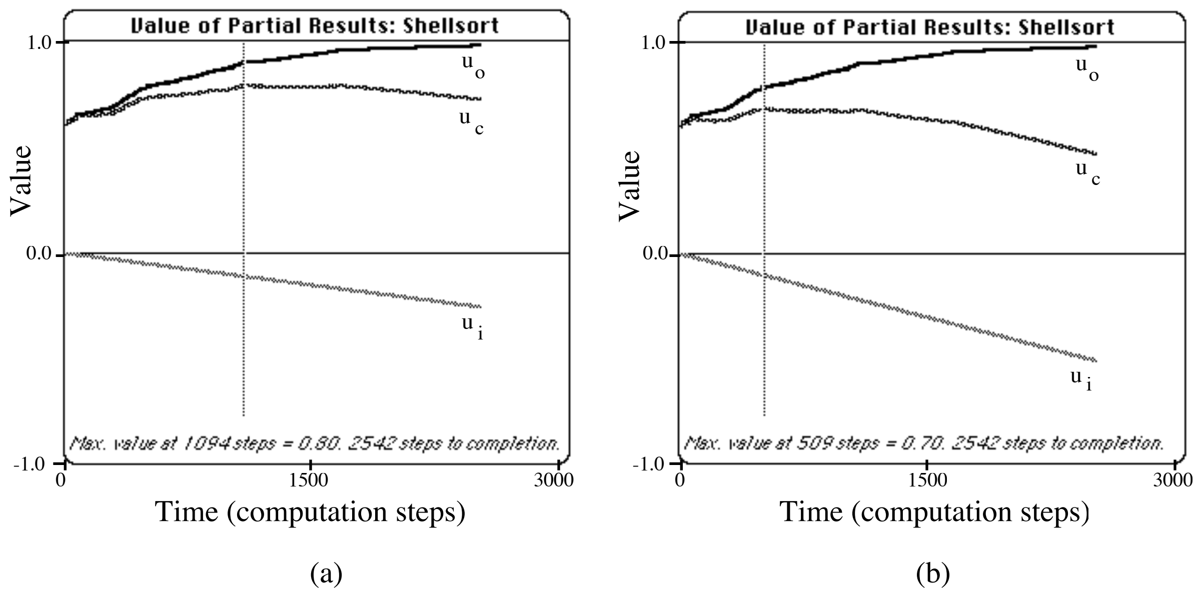


Figure 3.13: Economic analysis of ideal allocation of resources to shellsort.
(a) Shellsort's refinement of disarray for a randomized problem instance is modeled by a concave function. Thus, the ideal allocation of resource occurs in a region where the costs of reasoning are equal to the benefits of reasoning. (b) The ideal time for sorting a file can change depending on the urgency of the situation.

serious applications of normative metareasoning for sorting, and for other fundamental computational procedures, such as searching. In this vein, I shall describe several extensions of the work on partial computation for sorting.

We did not review the use of procedures for generating an EVC for selecting among alternative sorting algorithms or to reason in an a priori fashion about the ideal halting time for a single algorithm. Extensions of the utility-directed sorting work to EVC analyses would begin with the characterization of alternate algorithms, given a set of problem-instance features (e.g., size of instance and distribution of records in instance). Such characterization would require the assessment of probability distributions about the refinement of different attributes of value $p(v_i(\pi^{'}(I))|S, \pi(I), r, \xi)$. The assumption of parameterizations of named probability distributions might allow for efficient analysis. Given the speed of sorting, it is likely that difficult portions of the analyses would be performed offline, to generate policies about algorithm selection at run time. Such policies might be engineered to take advantage of run-time information about the distribution of records in a problem instance, as determined from evidence about the origin of a file, or gleaned from a brief, precursory phase of stochastic sampling of records from the file.

In another extension if this work, the ability of incremental-refinement strategies to make intermediate problem-solving states available can be useful for creating new policies from sequences of strategies (e.g., apply selection sort to bolster low-end completion efficiently and shellsort to refine the bounds on disorder). A custom-tailored sequence of strategies for generating $\phi(I)$ or $\pi(I)$ will often have greater computational utility than do more general, predefined policies. Thus, offline analysis of the efficiency of alternate sequences of partial-result "baton passing" could lead to new integrated algorithms that are ideal for a situation, given the cost of delay, the preferences of the person querying a computer for information, and the nature of the problem instance.

We can introduce even more flexibility into reasoning by moving the level of analysis from strategic to structural control, to consider control opportunities at the microstructure of computational activity. Although this task is more complex, the finer patterns of computation and control possible may enable a reasoner to generate more ideal refinement trajectories in the multiattribute utility space. Such research may also elucidate the control strategies implicit in familiar policies and stimulate the creation of more general, decision-theoretic strategies that could implement the familiar policies as special cases. That is, we may find that alternate sorting or searching algorithms can be viewed as making implicit assumptions about nature of the problem instance, the costs of computation, and the computer user's preferences.

Identifying useful dimensions of utility in computation and examining the refinement of partial results as a function of invested resources can also direct attention to new classes of approximation. For example, there is opportunity for developing inexpensive strategies for transforming valueless, intermediate states of traditional inflexible all-or-nothing algorithms into valuable partial results or into states that can be handed-off to other methods by a control reasoner. For example, in the realm of sorting, such techniques could be useful for concatenating $O(n \log n)$ strategies, in reaction to a specific problem instance, intermediate states, or observed real-time problem-solving trends.

3.5 Summary

Experimentation and analysis of the multiattribute structure of value in partial sorting results highlights several issues about reasoning under varying and uncertain resource constraints. First, we can demonstrate that interesting dimensions of value in partial results have been overlooked by computer scientists; more attention has been directed on techniques for computing a pretargeted goal. There is value in exploring the multidimensional structure of partial-result strategies. Also, we showed that the selection of a new strategy or the decision to cease computing, can be sensitive to details of the timewise-refinement trajectories, to the object-level utility function, and to the uncertainties in the functions describing the cost and availability of reasoning resources.

We shall now move beyond the definition of useful concepts and illustrative examples, and shall investigate the original motivation for normative metareasoning: the control of complex decision-theoretic inference. In Chapter 4, I shall focus on normative metareasoning about belief and action. I shall extend the general work on partial results specifically to partial results for inference. In Chapter 5, I shall present work on a new flexible inference algorithm. Then, in Chapter 6, we shall describe the operation of a system, named Protos, that uses normative metareasoning to control probabilistic inference. As we shall see, Protos applies the basic principles described in Chapters 2 and 4, to determine the ideal amount of time to devote to refining a problem before taking action in the world.

Chapter 4

Inference Under Bounded Computational Resources

In Chapter 2, I introduced partial results and reviewed principles of normative metareasoning for controlling the computation of these results. In this chapter, we shall examine the problem of computing beliefs and actions under bounded resources. I shall show how we can apply, in real-time, a small portion of reasoning resources to make decisions about the nature and extent of probabilistic inference. The normative control of automated decision analysis shall be posed as a model of rational action under resource constraints. I shall define the EVC in terms of the likelihood of future probability distributions over the truth of propositions about the state of the world. We shall make use of knowledge in the form of incomplete characterizations of the progression of probabilistic inference. EVC analyses, based on such knowledge, can tell us about the value of continuing to reflect about a problem versus that of taking immediate action in the world. In Chapters 6 and 7, we shall apply the normative-metareasoning techniques in the Protos system. Protos generates custom-tailored approaches to inference problems, under different time criticalities. We shall investigate how such flexibility can be useful in situations characterized by uncertain

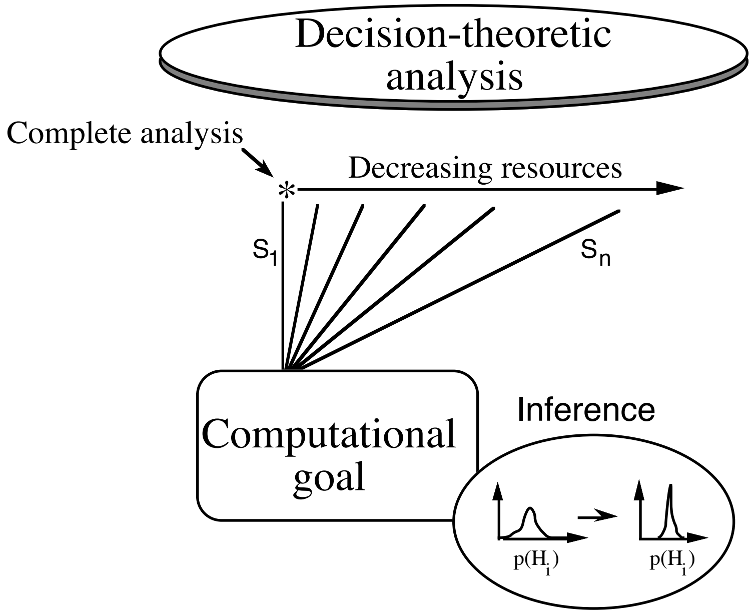


Figure 4.1: Reflective decision-analysis as a model of rationality.
We examine metalevel decision analyses for determining the ideal amount of time to perform inference about an object-level problem, considering the decision stakes, costs of delay, and knowledge about the likelihood of future probability distributions obtained with computation.

deadlines and challenges.

4.1 Building Versus Solving Decision Models

Two key phases of decision analysis are (1) the formulation of a decision problem, and (2) the solution of that decision problem. The formulation of a decision model includes the determination of possible actions and outcomes, the specification of a probabilistic model that relates the likelihood of outcomes to actions, and the specification of a preference ordering over outcomes. A general computer-based reasoner for performing automated decision analysis under varying resource constraints might be expected to perform the formulation and the solution of a decision problem. Thus, in principle, reasoning about beliefs and actions under bounded resources entails an analysis of the ideal control of processes for formulating a decision model, as well as for drawing conclusions by performing inference upon that model.

The formulation of problems that serve as the basis for inferential analysis has been considered the most ill-characterized phase of machine reasoning (Simon, 1972). Several investigators have speculated that there can be no principled machinery for problem formulation (Simon, 1973a; Buchanan, 1966). Decision analysts have referred to the construction of decision problems as the *framing* of a decision. Framing includes the identification and structuring of relevant distinctions and dependencies. Investigators have considered framing to be a complex task that requires human insight and domain knowledge. Thus, decision scientists have focused their analyses on the development of machinery for refining models *created by people*. There has been little progress on principled and robust approaches to the construction of decision models. Research to date on problem formulation, at the crossroads of AI and DA, includes the work by Holtzman on the rule-based manipulation of decision-problem templates (Holtzman, 1989), the use of production rules of fundamental relationships (Breese, 1990), the application by Wellman of a qualitative dominance analysis of important tradeoffs (Wellman, 1988), and the work by Heckerman and Horvitz on the reduction of a large decision model (Heckerman and Horvitz, 1990).

We shall not investigate the ideal control of the formulation of decision problems. Instead, to build decision models, I shall instantiate preexisting influence diagrams and belief networks with specific decision and utility information, and shall examine the use of normative metareasoning for solving decision problems under time pressure. Nevertheless, the principles of normative metareasoning, described in this dissertation, may be applied to the guidance of model-construction processes. Ideal control of the model formulation and model-solution phases of decision analysis awaits the development of robust problem-formulation machinery. We shall address this issue in more detail in Chapter 9.

4.2 Time-Dependent Utility

Let us explore concerns that arise in automated decision making under scarce resources. The graph in Figure 4.2 depicts an object-level influence-diagram representation of

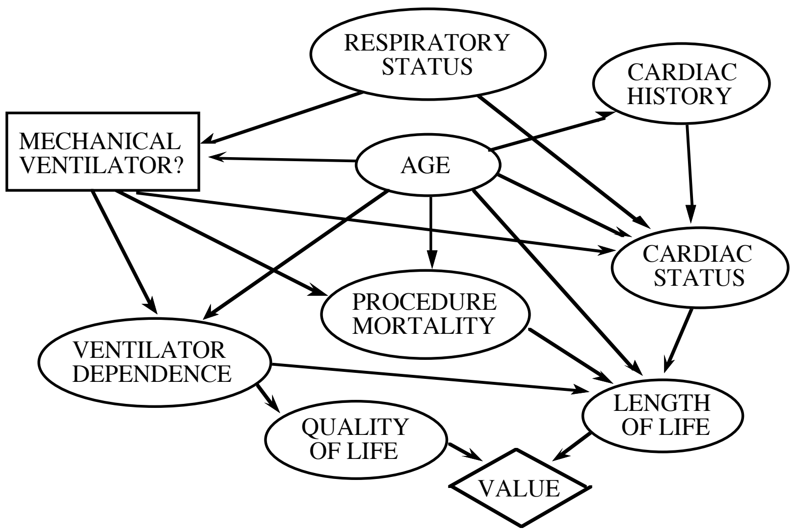


Figure 4.2: An influence diagram for an ICU decision problem.
The square node represents the possible actions. The diamond represents the utility of alternative outcomes. The oval nodes represent uncertain propositions. In this case, we are interested in the costs and benefits of talking action to assist a patient with breathing. We must consider the uncertainty in our knowledge about the respiratory status of the patient.

a time-pressured problem that might face an automated physician's assistant: A 79-year-old woman in the intensive-care unit (ICU) suddenly shows signs of breathing difficulty. The patient may be merely showing signs of stable *respiratory distress*, or may be facing imminient *respiratory failure*, the potentially fatal collapse of effective respiration. In this context, the primary decision is whether or not to recommend that the patient be placed on a mechanical ventilator. The decision (square node) depends on the value of respiratory status, which, in turn, depends on the probabilities of propositions in a large belief network serving as a medical knowledge base. The large oval nodes in the base decision problem represent uncertain states associated with placing an older person on a ventilator. The diamond represents the utility associated with different outcomes. Factors to consider in a decision to act include the possibility that the patient's breathing may become fatally blocked during insertion of a ventilation tube, and that it may take a long time to wean a patient from a ventilator; a patient may face a long hospital stay and be placed at high risk of mortality from a disease such as pneumonia after being placed on a respirator. However, if a patient turns out to be heading into respiratory failure, and is not placed on the ventilator immediately, she faces a high risk of cardiac arrest based on the disrupted physiology associated with abnormal blood levels of oxygen and carbon dioxide.

As indicated in Figure 4.3, we assume that the crucial probability of respiratory failure, given a set of observations, is made available to a computer-based reasoner only through inference in a complex belief network. Belief-network algorithms compute the conditional probability of propositions, given observations. A closeup of this network is displayed in Figure 4.4. This experimental 37-node belief network, named ALARM (Beinlich et al., 1989), represents distinctions and probabilistic relationships that are important in ICU medicine.

4.2.1 Actions and Outcomes in the World

In our example, there are only four outcomes. The patient either is in respiratory failure (H_1) or is not in respiratory failure (H_2), and we either will commit to assisting the patient with ventilation (A_1) or will not do so (A_2). Thus, we may erroneously decide not to treat a patient who is suffering from respiratory failure (A_2, H_1), we

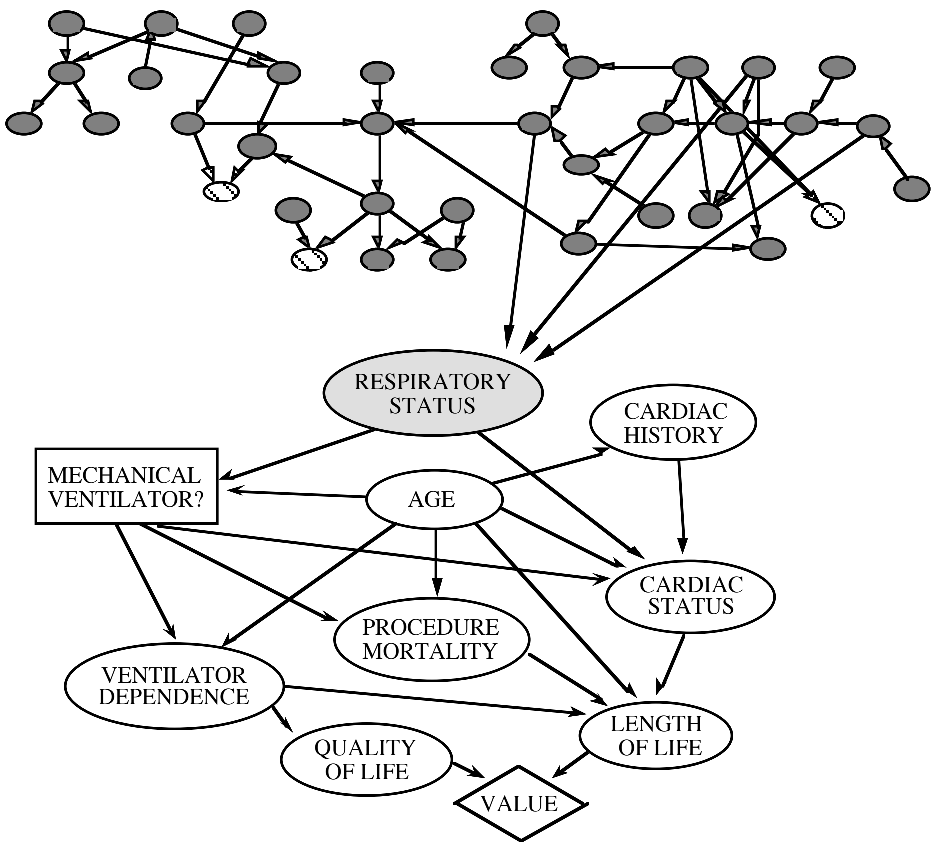


Figure 4.3: Computing a relevant probability with a belief network.
In response to observations, a computer-based reasoner applies a probabilistic-inference algorithm to a belief network (graph above influence diagram). The inference algorithm can be used to compute the conditional probability of one or more states of interest, given the observed findings (represented as lighter belief-network nodes). In this case, we are interested in the status of respiratory failure.

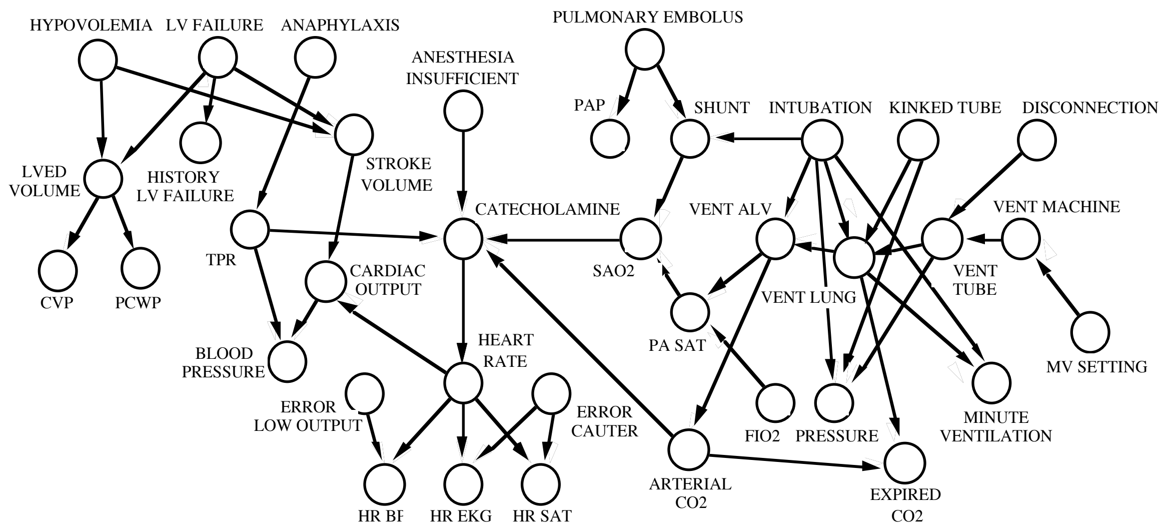


Figure 4.4: A multiply connected belief network for medical diagnosis.
This belief network represents the uncertain relationships among important propositions used in reasoning about the care of patients who are in an ICU. (See Appendix B for a legend to abbreviations in this belief network.)

may correctly treat a patient who is suffering from respiratory failure (A_1, H_1), we may erroneously treat a patient who is not suffering from respiratory failure (A_1, H_2), or we may correctly forego treating a patient who is not suffering from respiratory failure (A_2, H_2). Given the utility of different outcomes, we can compute the expected value of taking different actions A in terms of the likelihood of alternative outcomes H.

4.2.2 Assigning Utility to Outcomes

A system for reasoning about high-stakes medical decisions must assign utility to outcomes that are associated with a significant probability of death. Howard has detailed techniques for using decision analysis to assist people with decisions involving the possibility of severe health outcomes and death (Howard, 1980). Howard developed the *worth-numeraire* model to make possible the direct comparison of utilities for minor and major health outcomes. With the worth-numeraire model, utilities associated with major risks to life are measured in terms of life-and-death gambles.

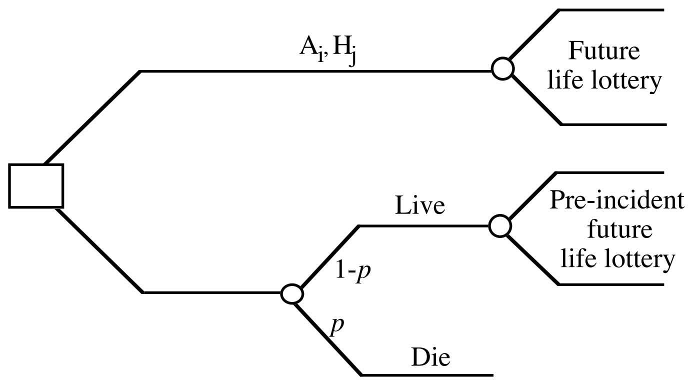


Figure 4.5: Assessing the utility of costly outcomes.
To assess the utility of an outcome A_i, H_i, we query a decision maker for the probability p of instant, painless death that would make him indifferent between continuing in his current situation (future life lottery) or having a $1 - p$ chance at continuing his life as if the acute incident facing him had not occurred (*pre-incident* future life lottery).

Utilities associated with minor diagnoses are measured in terms of dollars. The model determines what an individual would have to be paid to assume some greater probability of death, and what he would be willing to pay to avoid such a risk. A person is modeled as willing to pay a quantity of money that grows linearly in p for small risks of death ($p < 0.001$). The model suggests that, for small risks of death, people behave like expected-value decision makers, with some finite value assigned to their lives.

For significant risks of death, we would like our model to deviate significantly from a linear growth in the willingness to pay to avoid risk. Howard's model allows such deviation from linearity as the risk of death grows: As we would expect intuitively, the worth-numeraire model allows there to be some maximum probability of death, beyond which a person will accept no amount of money to be exposed to that risk of death.

To measure the utility of a major health disorder or misdiagnosis, we ask a decision maker to imagine that he is in a particular acute situation—say—acute respiratory failure, and that he is about to be treated as if he has respiratory distress. We then imagine that there is a treatment available that could rid the decision maker of the acute health incident instantly with probability $1 - p$, but that could kill him, immediately and painlessly, with probability p. If the patient accepts the treatment, and wins the associated lottery, we assume that he will continue his life-lottery as if the acute incident had not occurred. We assess from the decision maker the value of p that makes him indifferent between his untreated acute situation and the lottery associated with accepting the treatment. A decision tree that represents this decision problem is displayed in Figure 4.5. We take, as the utility of the current situation, the probability that the decision maker will win this lottery, or $1-p(\text{death})$. In time-critical life and death decision making, we can view the disutility of an outcome in terms of the probability of of death associated with that outcome. We assume that, if the patient does not die, he shall continue his life lottery, as if the acute incident did not happen.

Howard provides a means to convert utilities expressed in monetary terms to small probabilities of immediate, painless death. He describes the conversion of small probabilities of death to dollars in terms of dollars per micromort. A *micromort* is a one–in–1-million chance of immediate, painless death. Availability of a dollar per micromort conversion rate $v_{\mu\text{mt}}$ allows us to reason about alternate sources of risk in terms of dollar amounts.

4.2.3 Utility of Immediate Action

Returning to the respiratory patient, let us assume that an expert clinician—who has previously received consent from a set of patients to serve as the principal agent for a patient—has assigned the following utilities to the four outcomes: $u(A_2, H_2) = 1.0$, $u(A_1, H_2) = 0.7$, $u(A_1, H_1) = 0.4$, and $u(A_2, H_1) = 0.05$. In general, the expected utility of taking action A_i is

$$\text{eu}(A_i, p(H|E_k, \xi)) \;=\; \sum_{j=1}^{n} p(H_j|E_k, \xi)u(A_i, H_j) \tag{4.1}$$

where $p(H_j|E_k, \xi)$ is the conditional probability of state H_j, given observations E_k and implicit, or background state of information, ξ. In the context of computing probabilities with belief networks, ξ includes background information implicit in the structure and assessments of conditional probabilities of a belief network. Equation

4.1 expresses the notion that the expected utility of each action is determined by summing the utilities of each outcome, weighted by the probability of the outcomes.

In terms of our binary decision problem, we can reason about the utilities of the two actions in terms of the probability of respiratory failure, one of two mutually exclusive and exhaustive hypotheses about the physiology of the patient. Given observation of evidence E_k, the expected utilities of action (A_1) and of no action (A_2) are described by the following equations:

$$\mathrm{eu}(A_1, p(H_1|E_k,\xi)) = p(H_1|E_k,\xi)u(A_1,H_1) + (1 - p(H_1|E_k,\xi))u(A_1,H_2) \quad (4.2)$$
$$\mathrm{eu}(A_2, p(H_1|E_k,\xi)) = p(H_1|E_k,\xi)u(A_2,H_1) + (1 - p)(H_1|E_k,\xi))u(A_2,H_2) \quad (4.3)$$

The linear plots described by these equations, portrayed in Figure 4.6, intersect at a threshold probability of H_1 denoted p^*. The desired action (the decision with the highest expected utility) changes as the utility lines cross at p^*. A utility analysis dictates that a patient should not be treated unless a decision maker's belief in the truth of H_1 is greater than p^*. When a practitioner's belief falls below p^*, it is better to not treat the patient. When his belief rises above p^*, he should take action to treat.

4.2.4 Costs of Inference-Based Delay

In Section 4.2.3, we considered outcomes, and the utilities associated with alternative outcomes to be independent of time. The example of a patient gasping for breath, facing the risk of a long hospitalization or a cardiac arrest depending on our decision, poignantly demonstrates the significance of time-dependence of the utility outcomes in a high-stakes situation. Let us now integrate explicit knowledge about time-dependence of the utility of outcomes, and of the process of reasoning, into the decision problem. In answer to a query for assistance, an automated reasoner might have to propagate observed evidence about a patient's symptomology through a complex belief network. The results of approximate probabilistic inference can be expressed as some probability distribution over a *final* probability–the value that the computer would calculate with a belief network, given sufficient time to finish an

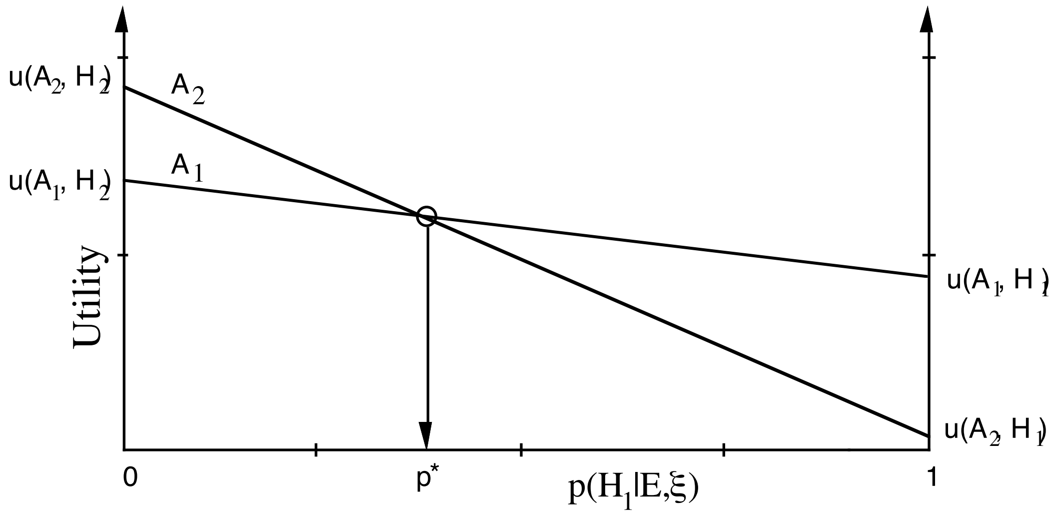


Figure 4.6: A graphical representation of the utility of two actions under uncertainty. The lines indicate the utility of action A_1 and action A_2 as a function of the probability of hypothesis H_1, $p(H_1|E_k, \xi)$. The lines cross at a probability of hypothesis H_1, called p^*. At this probability, the optimal decision changes.

exact computation. Assume that our reasoner may apply one of several incremental-refinement algorithms that can iteratively tighten the distribution on the probability of interest over time. We wish the system to make a rational decision about whether to make a treatment recommendation immediately based on a partial analysis, or to defer its recommendation and to continue to reason and thus to refine the analysis, given its knowledgeabout the costs of time needed for computation.

In critical settings, the utility associated with alternate actions can change with time. Let us redefine our decision problem in terms of the time an action is taken. We assume that the utility of correctly treating a patient threatened by a pathophysiologic state depends on the length of time a patient remains in the state before treatment. This assumption frequently is valid for severe health problems. For example, the utility of acting to treat respiratory failure can depend on how long the patient has been in failure. To consider the cost of a delay in action, a reasoner must determine how long a challenge has been present when the problem first comes to the system's attention. For our example, we shall assume that the presentation of respiratory

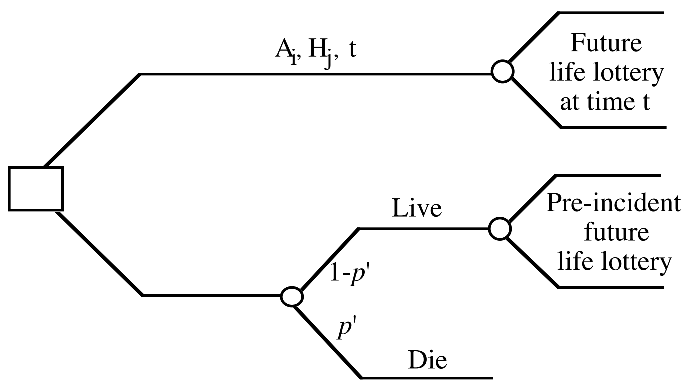


Figure 4.7: Assessing time-dependent outcomes.
We use A_i, H_i, t to refer to the outcome of action taken when state H_j has been true for time t. For assessing the utility of this outcome, we query a decision maker for the probability p' of instant, painless death that would make him indifferent between his future life lottery *at time* t after H_j becomes true or having a $1 - p'$ chance at continuing his life as if the acute incident facing him had not occurred.

symptoms occurs at the initiation of the pathophysiologic state, and that our reasoner begins its computational analysis of the problem at this time, t_α. In the more general case, a reasoner must consider uncertainty in t_α given some set of observations.

We represent the cost of delaying treatment, when that treatment is needed, by considering a continuum of decisions to treat at different times t after the initiation of a state. Let us use (A_i, H_i, t) to refer to the outcome defined by taking action when state H_j is true for time t. Figure 4.7 displays the lottery for assessing the utility of an outcome at progressively later times. To assess the utility of (A_i, H_i, t), we query a decision maker for the revised probability p' of instant, painless death that would make him indifferent between his future life lottery at the new time t or having a $1 - p'$ chance at continuing his life as if the acute incident facing him had not occurred. Rather than assess a utility for each time of action, we can assess functions that describe the cost of delay in terms of expected loss of dollars or increasing risk of instant, painless death. In work with an expert intensive-care physician, I found it useful to assess the change of utility of an action in a time-pressured setting in units of micromorts per second or *micromort flux*. In the case of the respiratory decision problem, the primary source of cost with delay is the increasing probability of cardiac arrest as a function of the time that we delay therapy.

For the respiratory problem, at some time t, the expected utility of acting in the presence of respiratory failure reverts to the utility of not acting at all. We substitute the static equation and functions for the expected utility of action defined previously in Equation 4.2, with a time-dependant equation:

$$\mathrm{eu}(A_1, p(H_1|E_k, \xi), t) = p(H_1|E_k, \xi)u(A_1, H_1, t) + (1 - p(H_1|E_k, \xi))u(A_1 H_2, t) \quad (4.4)$$

where $u(A_1, H_1, t)$ reverts to $u(A_2, H_1, t_\alpha)$ as some function of time t. In this example, we have assumed that delay of action will not affect the utility of a patient who does not require the intervention. That is,

$$\forall\ t : u(A_1, H_2, t) = u(A_1, H_2, t_\alpha)$$

However, in the general case, any outcome can be a function of time. With the time-dependent utility function, we must consider p^* as $p^*(t)$, a decision threshold that changes with time.

The arrow at the right side of Figure 4.8 indicates that the utility of treating a patient in respiratory failure converges on the utility of *not* treating a patient in respiratory failure over time. Note that, as the utility of treating a patient in failure falls, p^* increases.

4.3 Normative Metareasoning for Inference

A more comprehensive decision problem for our computer-based reasoner requires us to consider explicitly the costs of reasoning. We represent the more global decision problem in Figure 4.9. As indicated by the network in the upper portion of Figure 4.9, a more complete representation of the respiratory decision problem includes knowledge about the costs and benefits of applying different inference strategies for different amounts of time. This influence diagram represents the metareasoning problem, described in Chapter 2. Rather than our computer-based agent's goal being to optimize the object-level value, it is to optimize the utility associated with the value node in the metareasoning problem, labeled u_c. The consideration of inference-related, and

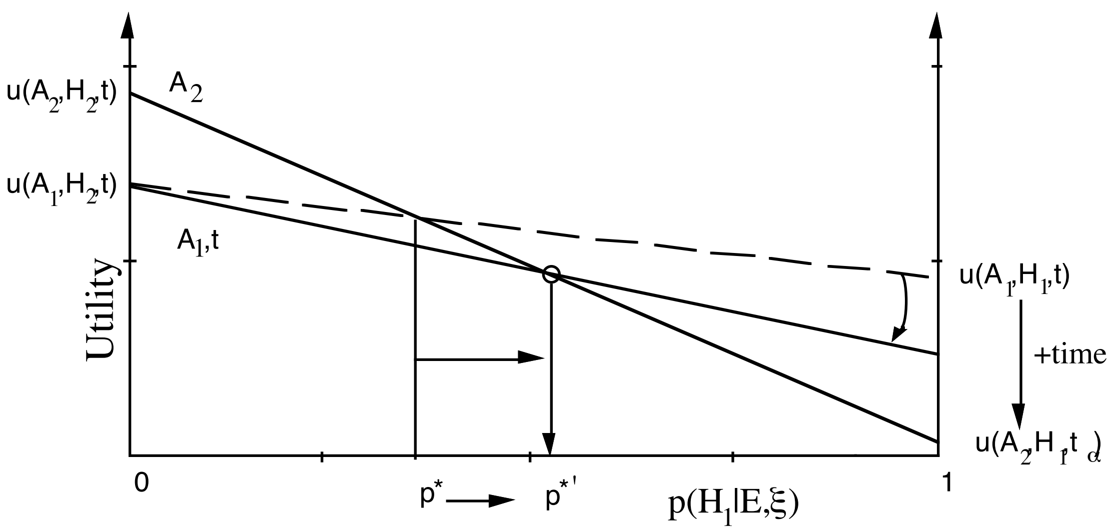


Figure 4.8: Representation of time-dependent utility.
This graph displays how the utility of an outcome can diminish as a function of time. In this case, the utility of outcome (A_1, H_1), taking action in the case of respiratory failure, decays to the utility of outcome (A_2, H_1), not taking action, as a function of time. Notice that the decision threshold, p^*, is also a function of time; in this case, p^* increases as the utility of (A_1, H_1) decreases.

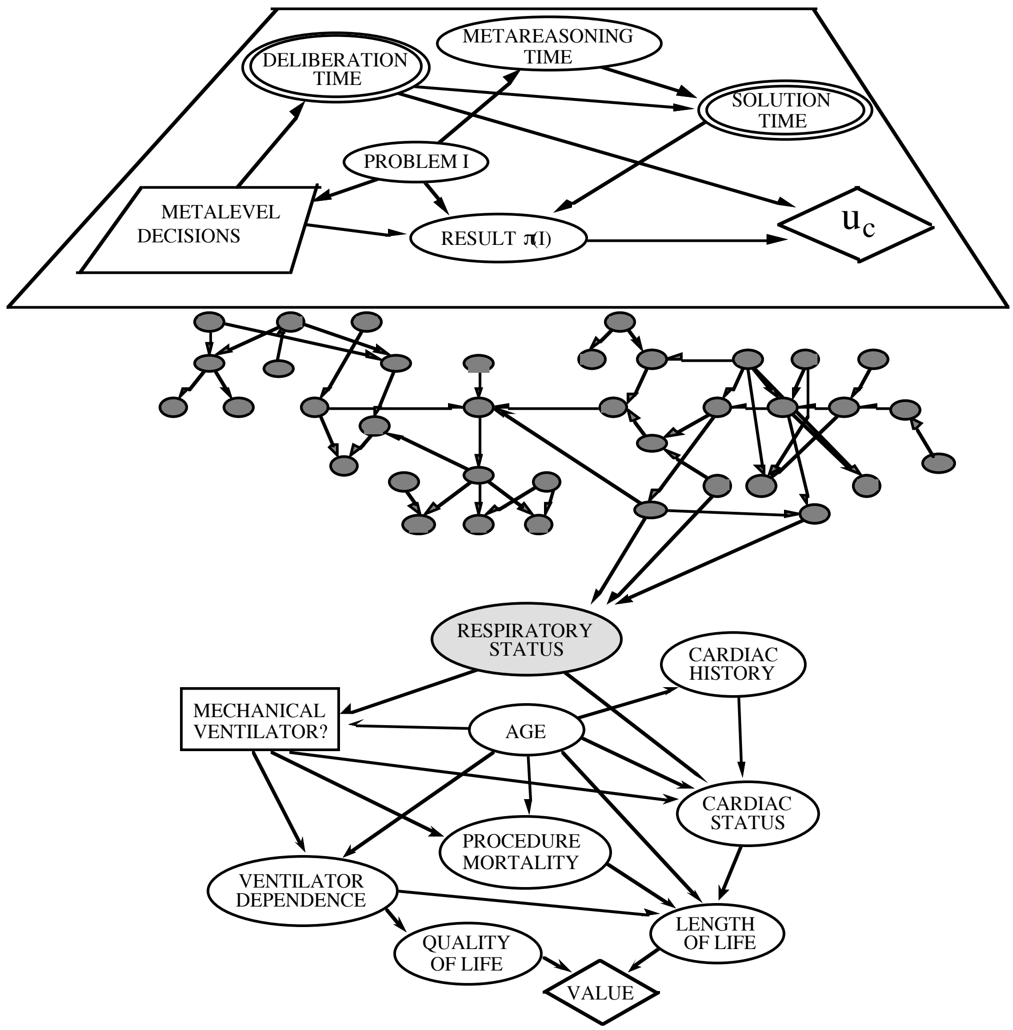


Figure 4.9: A more comprehensive analysis of a time-pressured decision problem. We add a level of reflective analysis to the computer-based decision problem by representing the costs and benefits of continuing to reason as well as representing the costs of the meta-analysis itself.

object-level components, of the comprehensive utility allows a reasoner to treat decisions and outcomes regarding the control of computation just like decisions about actions in the world. Decisions about strategies and about continuing to reason for different amounts of times before taking action can be viewed as additional decision alternatives.

In answer to a query for assistance, an automated reasoner must propagate the evidence through a complex belief network. The system makes a decision to apply one of several inference approximation algorithms that refines an approximation of a probability of interest. We shall address the decision about employing a strategy to reason for some quantity of time before halting with a recommendation. A sound metalevel decision requires us to analyze the EVC for alternative approximate inference schemes. Let us now examine this problem, and, in so doing, generate an equation for EVC for inference from the general statement of Equation 2.11 in Chapter 2.

4.3.1 Partial Results for Inference

Our computer-based reasoner's attention is focused on the calculation of $p(H_1|E_k, \xi)$, the *probability of respiratory failure*, given a set of observations E_k and background knowledge ξ. Let us place this computational goal in the framework introduced in Chapter 2. The problem instance facing the reasoner is to compute an answer to the query $p(H_1 = ?|E_k, \xi)$. We shall refer to such a target probability in a computational inference problem as $\phi(I)$, where the problem instance I is the query, and $\phi(I)$ is the exact probability that a computer-based reasoner would calculate if it had sufficient time to finish its computation. To simplify our notation, we shall use the shorthand of ϕ to refer to the $\phi(I)$.

Before an inference task is completed, our automated reasoner may be uncertain about the value of ϕ. Thus, partial results for probabilistic inference are *probability distributions that describe the current uncertainty about the final probability that would be calculated with sufficient time* to solve an inference problem completely. Flexible inference strategies incrementally tighten bounds or second-order probability distributions over a probability of interest before converging on the exact answer. We shall use $p(\phi)$ to represent partial results in inference problems to capture the essential notion that we are uncertain about an exact answer to a query.

Inference approximation strategies include methods for computing bounds for performing stochastic simulation.[1] These methods generate different classes of probability distributions, displayed in Figure 4.10.

Probability-bounding strategies generate partial results of the form of categorical (or deterministic) upper and lower *bounds* on point probabilities of interest (Cooper, 1984; Peng, 1986). Bounding techniques determine bounds on probabilities through a logical analysis of constraints acquired from a partial analysis. Bounds on the probability that would be computed with sufficient computation become tighter as additional constraints are brought into consideration.

Stochastic-simulation algorithms approximate a probability of interest by sampling the total probability space (Henrion, 1988; Pearl, 1987; Chavez and Cooper, 1989b). The methods generate a sequence of probability distributions over a set of states with ongoing computation. Some simulation algorithms produce distributions over a final result that is approximated by the binomial distribution. The variance with which the distribution converges on a probability with additional computation depends on the topology of the network, and on the nature of the probabilistic dependencies within the network. Another class of simulation algorithms, called *randomized approximation strategies* (RAS), produce partial results of the form of worst-case bounds on an error of a point probability estimate. RAS results take the form of inequalities on error bounds on an ideal result of the form *the probability that the divergence of a partial result, or estimator,* $\pi(I)$ *is less than* α *from the final result* $\phi(I)$ *is greater than* δ. Additional computation can be applied to increase the probability that the result lies between the error bounds, or to tighten the error bounds.

4.3.2 Belief about Future Beliefs

Let us develop a language for characterizing the progress that can be made by applying an inference strategy. We have used $p(\phi)$ to describe the uncertainty about ϕ, at the *present* moment, t_o, typically some time after t_α. Computational agents and human decision makers typically gain access to new information about a query by

[1] We review probabilistic inference strategies in more detail in Appendix A.

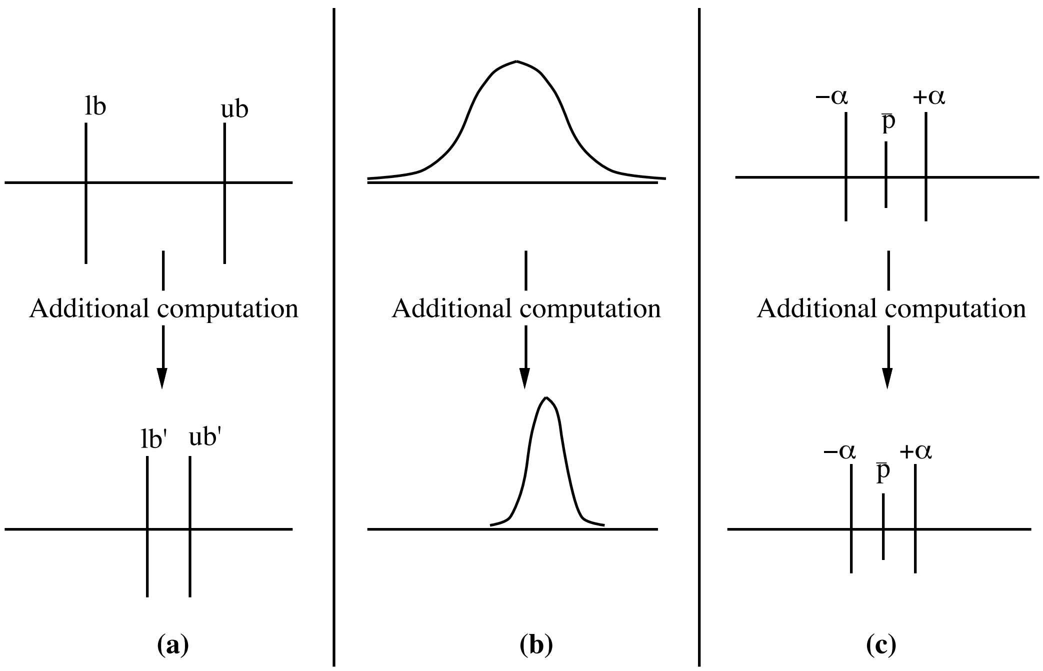


Figure 4.10: Families of partial results generated by computer-based inference. (a) With additional computation, some approximation methods generate categorical lower (lb) and upper (ub) bounds on a probability of interest, and work to tighten the bounds to produce new bounds lb' and ub.' (b) Other methods generate and tighten incrementally a probability distribution approximated by the binomial distribution. (c) Randomized approximation strategies (RAS) methods generate results that take the form of an inequality on the probability that an error α on a probability estimate $\bar{p}$ of the actual probability is greater than a constant δ. Ongoing computation can increase δ or decrease α.

continuing to refine a result. An agent may be uncertain about how a second-order probability distribution $p(\phi)$ will change with computation. It can be useful to decompose uncertainty about a probability into a *third-order distribution* over possible second-order distributions that might be obtained after performing some inference.

We use $p_t(\phi)$ to refer to a future second-order distribution at future time t. We refer to third-order distribution information about the current uncertainty about this future belief as $p(p_t(\phi))$. Such *belief-constellation* knowledge describes uncertainty in terms of a current probability distribution over a *set* of possible future second-order distributions. Because uncertain knowledge about future beliefs depends on the reasoning strategy S, we must, in general, index belief about future beliefs by the strategy being used. We use $p(p_t(\phi)|S, t)$ to refer to a distribution of probability distributions over ϕ after applying inference strategy S for time t.

Belief-constellation knowledge can take the form of uncertain or logical constraints on parameters that define families of future stereotypical or *named* probability distributions over a probability. I will describe the use of third-order distributions in EVC calculations for bounding algorithms in Section 4.4. In Chapter 5, I will present an algorithm that generates belief-constellation knowledge, and demonstrate its application in EVC computation in Chapters 6 and 7. An example of belief-constellation knowledge is a probability distribution that describes the interval between upper and lower bounds over ϕ after some computation with a probability-bounding algorithm. We shall explore how information about the current bounds interval, and future bounds interval obtained with additional computation, can *constrain* the $p_t(\phi)$. Knowledge about both classes of belief about future belief may be acquired through empirical analysis of a belief network, may be made available by an inference algorithm at run time, or may be proved theoretically.

4.3.3 Partial Results for Action

We have discussed classes of partial results for beliefs. What are partial results for decisions? We can consider partial results for decisions, given uncertainty about the probabilities of important propositions, to be actions in the world dictated by $p(\phi)$. If we are forced to act immediately, our best action is dictated by the mean of the distribution (Howard, 1970). We take action A that maximizes our expected utility, given the mean of $p(\phi)$, denoted $< p(\phi)>$. Note that the mean of $p(\phi)$ is

$$<\phi>= \int_\phi \phi \; p(\phi) \tag{4.5}$$

The utility of that best decision is equal to the utility of the action that would be taken if belief in ϕ had been a point probability at the mean of $p(\phi)$. That is,

$$\arg\max_A \mathrm{eu}(A, p(\phi), t_o) = arg\max_A \mathrm{eu}(A, <p(\phi)>, t_o)$$

4.3.4 Expected Value of Perfect Computation

What if our reasoner could compute an exact answer to a query for a relevant probability instantaneously? How valuable would this reasoning be? To answer this question, we introduce the *expected value of perfect computation* on ϕ, denoted by EVPC_ϕ. Suppose that, after reasoning for a few milliseconds, an automated reasoner has generated a probability distribution over ϕ. The EVPC_ϕ is the value of instantaneous complete computation of a target probability in a decision setting. Instantaneous complete thinking would collapse the current probability distribution over ϕ into an impulse (i.e., ϕ would be known with certainty). Thus, we say that EVPI_ϕ is equal to the value of *clairvoyance* about the final state of the computer. Unfortunately, real-world computers do not have the ability to be clairvoyant about the end result of reasoning.[2]

Given the current probability distribution $p(\phi)$, we define EVPC_ϕ as follows:

$$\mathrm{EVPC}_\phi \;=\; \left[\int_\phi p(\phi) \max_A \mathrm{eu}(A, \phi, t_o)\right] - \max_A \mathrm{eu}(A, <p(\phi)>, t_o) \tag{4.6}$$

where $\max_A \mathrm{eu}(A, <p(\phi)>, t_o)$ is the utility, associated with the best action A, based on taking an *immediate* action using the current mean belief, $<p(\phi)>$. This measure tells us that the value of computing the final answer is just the difference in utility between the current best action and the summation of the best actions weighted by the probability of different final beliefs.

[2]Theoretical computer scientists have used the notion of an omniscient *oracle* as a tool in the analysis of algorithms.

4.3.5 Value of Computation for Inference

Real computers rarely deliver the full EVPC on difficult problems because they must expend valuable resources to reason. We shall now examine the EVC for inference. The EVC analysis for inference follows from the general principles elucidated in Chapter 2. The EVC of an inference method depends on the nature of uncertain or partial knowledge that can be used to reason about what $p(\phi)$ will be after computation for time t.

Let us examine the EVC as a function of $p(\phi)$ and $p_t(\phi)$. We can use these two quantities to adapt Equation 2.11 in Chapter 2 to inference, with $\pi(I) \rightarrow p(\phi(I))$ and the inclusion of a decision variable A:

$$\begin{aligned}\mathrm{EVC}(S,t) \;=\; & \int_{p_t(\phi)} p(p_t(\phi)|S,t) \int_{\phi} \max_A \mathrm{eu}(A, p_t(\phi), t) \times p(p_t(\phi)|S,t) \\ & - \max_A \mathrm{eu}(A, p(\phi), t_o) \end{aligned} \tag{4.7}$$

In Equation 4.7, we sum over the new probability distributions on ϕ expected at time t, weighted by the current belief, $p(p_t(\phi))$, that thinking with some strategy S for time t will lead to each of the revised distributions, $p_t(\phi)$.

If we assume that we shall have to act immediately after computing for t seconds, we can simplify Equation 4.7 by reformulating that equation in terms of the mean of ϕ. Thus,

$$\begin{aligned}\mathrm{EVC}(S,t) \;=\; & \int_{p_t(\phi)} p(p_t(\phi)|S,t) \max_A \mathrm{eu}(A, <p_t(\phi)>, t) \\ & - \max_A \mathrm{eu}(A, <p(\phi)>, t_o) \end{aligned} \tag{4.8}$$

To add a consideration of metareasoning time for the calculation and optimization of EVC, we decrease the time available for updating the probability distribution by the cost of the EVC-based optimization, $r^{\mathcal{M}}$. Thus,

$$\begin{aligned}\mathrm{EVC}^{\mathcal{M}}(S,t) \;=\; & \int_{p_t(\phi)} p(p_t(\phi)|S,t') \max_A \mathrm{eu}(A, <p_t(\phi)>, t) \\ & - \max_A \mathrm{eu}(A, <p(\phi)>, t_o) \end{aligned} \tag{4.9}$$

where $t' = t - r^{\mathcal{M}}$, the time delay diminished by the time $r^{\mathcal{M}}$ required for the computation of 4.9.

We can substitute Equation 4.9 into the computation-optimization formulae of Section 2.5 to select the best strategy, and the length of time that strategy should be applied, given uncertain distributions about how probability distributions will change with computation. Such a global analysis is complex. Instead, we shall explore the use of a myopic or greedy approximation of EVC for small constant amounts of computation time, T. A tractable myopic analysis allows us to continually recompute the EVC during inference. Under uncertainty, relying on an a single a priori analysis of the EVC after a long delay for computation requires us to throw away detailed information about future belief made available *during inference*. We would like to have the ability to incorporate new information about ϕ generated during inference incrementally, to condition knowledge about future belief on the most recently computed $p(\phi)$. For applying the greedy formulation of the EVC, we compute the EVC for a quantity of time for each candidate algorithm and select the algorithm with the greatest EVC.

As displayed in Figure 4.11, our goal is to develop tractable EVC-evaluation machinery for use in real-time incremental EVC analyses. We shall examine examples of the greedy application of Equation 4.9 for the construction of such EVC evaluators. In an approximate incremental analysis, when the cost of computation (embodied in our comprehensive utility function) becomes greater than the benefit of computing ($\mathrm{EVC}(S, t) \leq O$) for any available algorithm, a computer reasoner should cease reflecting about a problem and should take (or recommend) action in the world.

4.4 EVC for Probability Bounding: A Constraint-Based Analysis

I have investigated tractable EVC analyses of bounding and simulation algorithms. We shall review the derviation of a tractable greedy EVC analysis for probabilistic inference algorithms that post upper and lower categorical bounds on a probability of interest. I call the special case of EVC, the EVC/BC, for *bounds categorical.* In Chapter 5, I shall describe a new flexible probability-bounding strategy named *bounded conditioning* that can be analyzed with EVC/BC.

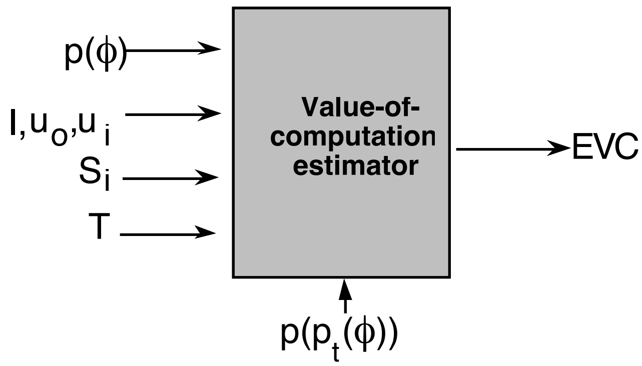


Figure 4.11: Toward tractable EVC estimators for controlling probabilistic inference. We wish to develop tractable EVC-estimation machinery, that reports the EVC in response to (1) a problem-instance (I), (2) object-level and inference-related components of utility (u_o, u_i), (3) the current uncertainty about probability of interest $p(\phi)$, and (4) a quantity of computation time (T). Such EVC estimators use uncertain knowledge about future probability distributions, indicated by the quantity, $p(p_t(\phi))$, at the bottom of the figure.

Let us turn back to our ICU problem. Assume that our automated reasoner, facing the challenge described in Section 4.2, has applied a probability-bounding algorithm, and has computed upper and lower bounds on ϕ, with an upper bound at ub and lower bound at lb. Let us assume that our reasoner has no information about where ϕ is within the bounds—except that the final computed result will be between the current bounds. We shall perform a myopic EVC analysis, given only one piece of information that is made available to our reasoning system at each step of the myopic analysis: We know how tight the interval shall be on future bounds obtained after computation for an additional time t. What is the value of continuing to compute, given only information about the current bounds and about the new bounds interval?

If we have no information except for the constraints dictated by the current bounds, we can model our knowledge about ϕ with a uniform distribution between the bounds.[3] Figure 4.8 shows the lower and upper bounds and the mean of the uniform distribution (or any distribution that is situated symmetrically between the bounds). Recalling the decision rule from Equation 4.3.3, if an immediate decision is

[3]We can weaken this assumption in the EVC/BC analysis by assuming only that $p(\phi)$ is some symmetric distribution between the bounds. Detailed knowledge about convergence would further specify this distribution.

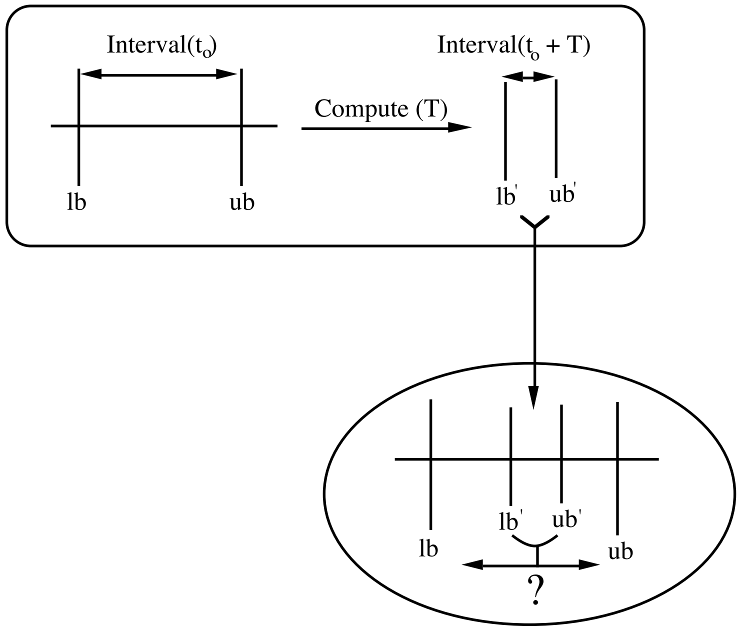


Figure 4.12: The probabilistic-bounds EVC problem.
This schematic captures the problem statement: Given the current bounds (lb,ub) calculated by an automated reasoner and the additional convergence expected with computing for some additional time T, what is the value of the additional computation? To calculate the EVC, we must reason about the expected location of the tighter bounds (lb', ub') within the current bounds.

forced, a reasoner should take the action with the greatest utility, given the mean of the distribution. Because the current mean is greater than p^*, in this case, the best action is A_1.

Let us explore how a system can make use of certain or uncertain knowledge about how bounds converge with computation to calculate an EVC of continuing inference versus halting and acting with the best decision available. In our myopic analysis, we assume that we shall make a decision after computing for t additional seconds. Our decision at that time will depend on the value of the future mean $<p_t(\phi)>$. We shall compute the value of computation by considering the ideal decision after t seconds for each possible value of the future mean of $p_t(\phi)$, $<p_t(\phi)>$.

At the present moment, we know the current upper and lower bounds on ϕ. Assume that one additional piece of information is revealed to us: We are told the value of a future, tighter interval between the upper and lower bounds on ϕ that will result from additional inference. Although we now know the size of the future interval, we do not know the location of the future upper and lower bounds. Given no other information, we assume that the new bounds may be found with equal likelihood at any possible position within the current bounds, and that when we compute the tighter bounds, ϕ will again be uniformly distributed between the new bounds. As indicated by Figure 4.12, given information about an interval, we must consider all possible configurations of the new bounds given the current constraints. We consider the value of computing the tighter bounds by moving the interval uniformly within the current interval, and considering all values of the mean of the future distribution. Certain or uncertain information about a new bounds interval is belief-constellation knowledge: It dictates a set of revised distributions. To compute an EVC, we shall sweep the new bounds through all positions of the new distribution, and integrate the value of the best decision for each possible ϕ, weighted by belief in that ϕ.

Figure 4.13 displays a graphical analysis of the probabilistic-bounds EVC problem. As we sweep a set of bounds, separated by an expected future interval, over the current bounds interval, the mean of the future uniform distributions sweeps between positions within the current bounds. The figure shows that the mean of the distribution on ϕ, after computation for t, will range over an interval defined, at the lower end, by the sum of one-half of the new interval and the current lower bound, and, at the upper end, by a point the same distance below the current upper bound.

In the EVC calculation, we sum the distributions and the value associated with the best decisions over the new intervals for each position of the new bounds. When the mean is above p^*, we sum over the utility of acting for all states of belief greater than that threshold; when the mean is below p^*, we similarly consider the utility of not acting. Given our current bounds and a convergence fraction, we sum the utilities of the best decision at the future means and subtract the utility of the best action without additional computation. The EVC/BC formula is an approximation because of our single-step myopic assumption.

I shall first review several definitions useful in expressing formulae for EVC/BC. At the current time t_o, we have some upper and lower bounds (lb, ub) on a probability

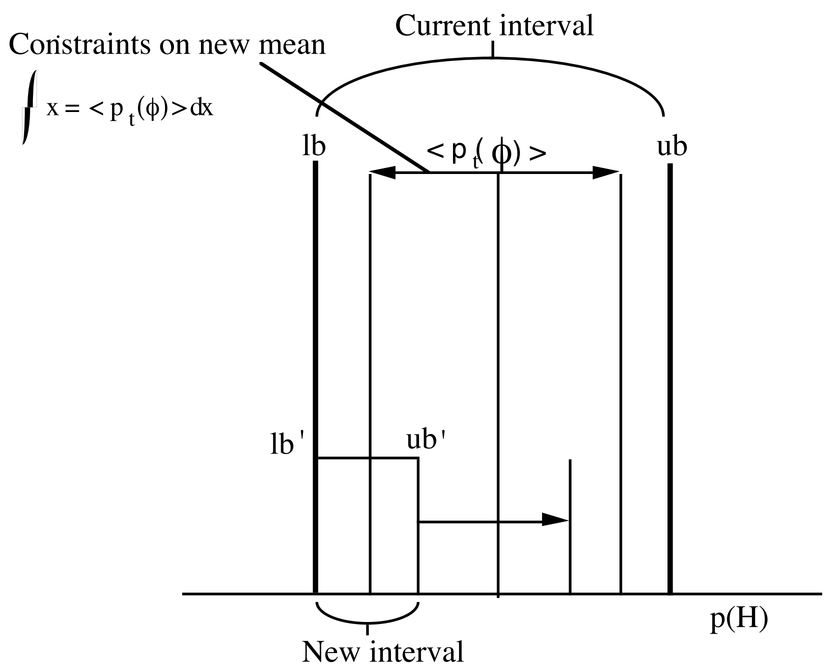


Figure 4.13: A constraint-based EVC analysis.

At the present moment, we know the current upper and lower bounds (lb,ub). We can use knowledge about a new interval between the revised upper and lower bounds (lb',ub') to compute an EVC. We sweep the new bounds through all positions of the new distribution. This figure graphically demonstrates that the mean of the distribution on ϕ, after computation for t, will range over an interval defined, at the lower end, by the sum of one-half of the new interval and the current lower bound, and, at the upper end, by a point the same distance below the current upper bound. We sum the distributions and the value associated with the best decisions over the new intervals for each position of the new bounds.

of interest. The current *bounds interval*, $int(t_o)$, is the difference, $ub - lb$. Assuming a symmetric distribution between the bounds, the current mean is

$$<p(\phi)>= \frac{lb + ub}{2}$$

Let us assume that we have certain knowledge of the form $\text{int}(t)$, the bounds interval after computing for some time t. Such knowledge imposes constraints on future second-order distributions. Let us consider the possible values of the mean of the future distribution, $<p_t(\phi)>$. As highlighted in Figure 4.13, we assume the mean will be distributed symmetrically between upper and lower *bounds on the future mean* (lm, um). We shall consider a greedy analysis in terms of a constant increment of time T. In terms of $\text{int}(t_o + T)$, the bounds on the mean of the future distribution are defined as

$$\begin{aligned} lm &= lb + \frac{\text{int}(t_o + T)}{2} \\ um &= ub - \frac{\text{int}(t_o + T)}{2} \end{aligned}$$

A simple subtraction between these equations reveals that the interval on the value of the mean, is just the difference between the old and new intervals. That is,

$$\begin{aligned} um - lm &= ub - lb - \text{int}(t_o + T) \\ &= \text{int}(t_o) - \text{int}(t_o + T) \\ &= \Delta int(t_o, t_o + T) \end{aligned}$$

Using these definitions,

$$\begin{aligned} \text{EVC/BC} = & \frac{1}{\Delta int(t_o, t_o + T)} \int_{lm}^{um} \max_A \text{eu}[A, <p_t(\phi)>, t_o + T] d<p_t(\phi)> \\ & - \max_A \text{eu}(A, <p(\phi)>, t_o) \end{aligned} \tag{4.10}$$

Let us now apply the EVC/BC to inference about a binary decision problem. We substitute into Equation 4.10 knowledge about the utility of different decisions. Referring to our original decision problem, as displayed graphically in Figure 4.8, we see that, as the probability of H_2 increases, the decision that dominates changes from A_2 to A_1 at $p^*(t)$.

Thus, Equation 4.10 can be rewritten as

$$\begin{aligned} \text{EVC/BC} &= \frac{1}{\Delta int(t_o, t_o+T)}[\int_{lm}^{p^*(t)} \text{eu}(A_2, <p_t(\phi)>, t) d <p_t(\phi)> \\ &+ \int_{p^*(T+t_o)}^{um} \text{eu}(A_1, <p_t(\phi)>, T+t_o) d <p_t(\phi)>] \\ &- \max_A \text{eu}(A, \left(\frac{lb+ub}{2}\right), t_o) \end{aligned} \quad (4.11)$$

I shall substitute information about the expected utility of treating and not treating a patient with a suspected ailment, and integrate. The expected utilities of these actions are described by 4.4 and 4.3. To simply the formulae for substitution, we use the following abbreviations:

$$\begin{aligned} W &= u(A_2, H_2) \\ X &= u(A_1, H_2) \\ Y &= u(A_1, H_1) \\ Z &= u(A_2, H_1) \end{aligned}$$

We can assume that any of the outcomes may be a function of time. The utility of acting and not acting, for the case of the respiratory problem, in terms of the mean of a future distribution at time $t_o + T$, is

$$\text{eu}(A_1, <p_t(\phi)>, t_o+T) = Y(t) <p_t(\phi)> + X(1- <p_t(\phi)>) \quad (4.12)$$

$$\text{eu}(A_2, <p_t(\phi)>, t_o+T) = Z <p_t(\phi)> + W(1- <p_t(\phi)>) \quad (4.13)$$

Although only Y is assigned a time-dependent utility function in our respiratory example, we shall develop a formula that allows the utility of any of the outcomes to be a function of time. By setting Equations 4.12 and 4.13 equal to each other, we can express $p^*(t)$ as a function of W, X, Y, Z:,

$$p^*(t) = \frac{W(t_o+T) - X(t_o+T)}{W(t_o+T) - X(t_o+T) + Y(t_o+T) - Z(t_o+T)} \quad (4.14)$$

Now, by substituting Equations 4.12 and 4.13 into 4.11, we have for theEVC/BC,

$$\frac{1}{\Delta int(t_o, t_o+T)}[\int_{lm}^{p^*(t_o+T)} Z(t_o+T) <p_t(\phi)> + W(t_o+T)(1- <p_t(\phi)>) d <p_t(\phi)>$$

$$+ \int_{p^*(t_o+T)}^{um} Y(t_o+T) <p_t(\phi)> + X(t_o+T)(1 - <p_t(\phi)>) d <p_t(\phi)>]$$
$$- \max_A \text{eu}\left(A, \left[\frac{lb+ub}{2}\right], t_o\right) \tag{4.15}$$

Integrating Equation 4.15, we have

$$\begin{aligned} \text{EVC/BC} = & \frac{1}{\Delta int(t_o, t_o+T)}[\frac{X(t)(p^*(t)^2 - lm^2) + Y(t)(lm^2 - p^*(t)^2)}{2} + Y(t)[p^*(t) - lm] \\ & + \frac{W(t)(um^2 - p^*(t)^2) + Z(t)(p^{*2}(t) - um^2)}{2} + Z(t)[um - p^*(t)] \\ & - \max_A \text{eu}\left(A, \left[\frac{lb+ub}{2}\right], t_o\right) \end{aligned} \tag{4.16}$$

where, for brevity, we use t for $t_0 + T$. The last term of Equation 4.16 depends on whether the mean of the current distribution is greater or less than $p^*(t_o)$,

$$\max_A \text{eu}(A, <p(\phi)>, t_o) = \begin{cases} X(t_o)(\frac{lb+ub}{2}) + Y(t_o)(1 - [\frac{lb+ub}{2}]), & (\frac{lb+ub}{2}) \leq p^*(t_o+T) \\ W(t_o)(\frac{lb+ub}{2}) + Z(t_o)(1 - [\frac{lb+ub}{2}]) & \text{otherwise} \end{cases} \Bigg]$$

Equation 4.16 assumes only one of the outcomes is time time-dependent. In general, we make any of the outcomes time-dependent.

Thus, equation 4.16 yields the EVC of inference as a function of

- The utilities for each of the four outcomes
- The current bounds on ϕ
- The convergence of bounds with time ($\Delta int[t_o, t_o + T]$)
- Functions describing the cost of delay for any of the outcomes

4.4.1 Myopic EVC/BC Approximation for EVC

Let us pause to consider the relationship between the myopic EVC/BC and more accurate, global measures of the expected value of computation. One area of interest centers on the consistency of parameterized descriptions of future probability distributions and the current probability distribution. When information about the size of a future bounds interval is revealed to an agent that previously had knowledge only about the current bounds on a probability of interest, the current distribution must be updated. For example, learning the value of future, tighter bounds, and considering these bounds to be uniformly distributed in positions allowed by the current bounds, updates the agent's previous uniform probability distribution to a trapezoidal distribution. To generalize our metareasoners so that they can use knowledge that is available about the iterative tightening of bounds, we need to identify conjugate distributions that can be used to describe and to make consistent the probability distributions within current and possible future bounds. The essential form of such desirable *reflective-conjugate* distributions would be invariant to the width of bounds. The current EVC/BC formulae may be viewed as approximate for making sound use of knowledge about a sequence of EVC analyses: if the future bounds are much smaller than the current bounds, the probability distribution implied by a uniform distribution over the future distributions approximates a uniform distribution.

There are also questions about the effectiveness of the myopic EVC/BC analysis to identify global EVC trends in real problems. Great nonmonotonicity in the EVC of reasoning can lead a metareasoner to halt computation when it reaches an EVC local minima. In Chapter 6, we shall explore this potential problem and describe alternate techniques for giving metareasoners the ability to peek over the valleys defined by EVC local minima. We shall have the opportunity to observe examples of EVC nonmonotonicity and monotonicity in the case analyses presented in Chapter 7.

4.4.2 Use of EVC/BC

The tractable EVC/BC formula (Equation 4.16) was formulated for binary decision problems that depend on the truth of a single proposition. We can identify relevant decisions of interest and apply the EVC/BC to the decision problem iteratively until we see the EVC/BC become zero or negative. For situations where we are concerned with whether to act now or to delay addressing a potential problem, we default to inaction and wait until the EVC/BC becomes zero or negative and the mean $<p(\phi)>$ is greater than p^* for the action. If EVC/BC becomes negative while the mean is less than p^* for the action, we continue to reason.

We can apply the EVC/BC more generally to sets of n binary decisions. For each possible fault or disorder of interest H_i (e.g., all the treatable faults in a belief network), we identify a treatment action A_i. In every cycle of EVC analysis, we consider in turn each binary decision problem. That is, for each hypothesis H_i, we determine the value of taking action (A_i) versus not taking action ($\neg A_i$) to address that fault. If the EVC/BC is positive for any H_i, we delay the corresponding action A_i, and continue to deliberate. If the EVC/BC becomes negative for any hypothesis and the mean of the probability distribution over that hypothesis is greater than p^*, we act with A_i. If the EVC/BC becomes negative for any hypothesis and the mean of the probability distribution for that hypothesis is less than p^*, we continue to keep the action under consideration. The appropriateness of considering n treatment actions as independent decisions depends on the interactions among the treatments. If we consider interactions among treatments to be negative in the general case, the p^* for each treatment decision is likely to be smaller than the decision threshold that would be derived in a more comprehensive decision analysis (Heckerman and Horvitz, 1990). Thus, a more comprehensive decision model will tend to perform more conservatively than the independent treatment approach.

The EVC/BC solution for binary decision problems also has applications in areas of decision making where a task is structured into a hierarchy of tasks and where one decision must be made before another. In such a tree of decisions, this approach can be used to make a decision about how long to dwell on one problem before moving on to the next decision.

We shall explore the use of the EVC/BC in the Protos system in Chapter 6. First, we shall examine the properties of a new probabilistic reasoning method, named *bounded conditioning*, developed as part of this thesis research. I shall show how we can use Equation 4.16 to evaluate iteratively the value of continuing to compute for a small amount of time, versus halting computation and acting with the best decision. At each step of the iterative EVC/BC analysis, the latest computed probability and knowledge about the likelihood of future probability distributions is considered by the system.

4.5 Relationship to Other Research

This dissertation research is distinguished from earlier studies of probability and utility in AI by its focus on the use of decision-theoretic principles to control the decision-theoretic reasoning of limited agents that must act under constraints in computational resources. I introduced the notion of pursuing bounded optimality—optimizing the design, reasoning, and actions of agents with explicit consideration of the limitations in an agent's computational abilities and resources—to highlight the motivation and long-term goals of this research (Horvitz, 1986; Horvitz, 1987c; Horvitz, 1987b). My study of the control of probabilistic reasoning (Horvitz et al., 1989a) and the use of multiattribute utility to evaluate and control basic computational processes (Horvitz, 1988) comes in the context of other previous and ongoing efforts to explore the use of decision theory in automated reasoning.

The use of decision theory for the control of reasoning was discussed over a decade ago by the statistician I.J. Good in speculation about an effective way to control game-playing search. Good had earlier discussed the explicit integration of the costs of inference within the framework of normative rationality, defining *type I* rationality as actions that are consistent with the axioms of decision theory, regardless of the cost of inference, and *type II* rationality as behavior that takes into consideration the costs of reasoning (Good, 1952). Good described how a chess playing machine might someday be based on knowledge about how belief in the value of alternative moves would change with computation (Good, 1977). Related work on the richer notion of rationality that includes the cost of reasoning has been performed by several investigators in the decision science community. Logan (Logan, 1985) and Heckerman and Jimison (Heckerman and Jimison, 1989) have explored the costs and benefits of expending additional effort in assessing probabilistic quantities from people for use in decision analyses. Matheson and Watson have explored the use of knowledge about how the quality of a decision analysis might change with the expenditure of additional computational or modeling effort (Matheson, 1968; Watson and Brown, 1978).

Several AI investigators have previously examined the use of probability and utility to control reasoning. Barnett examined the use of probability and expected utility to reason about the value of control for ordering the application of search operators in a problem solver (Barnett, 1984). Shortly after Barnett's work, Smith and Genesereth, explored the use of decision theory for selecting alternative logical reasoning strategies (Smith, 1986). In this work, the ordering of conjunctions of propositions in a theorem-proving system is controlled by uncertain estimates of the size of search problems associated with alternate orderings. Several more recent studies on decision-theoretic control of search algorithms have been undertaken during the course of this dissertation research. Russell and Wefald have studied the interpretation of heuristic measures used in search as utilities. They elucidated several assumptions about utility in search and explored how utility estimates can be employed to make decisions about the value of taking alternative paths through a search tree (Russell and Wefald, 1989). Hansson and Mayer have explored the promise of performing probabilistic inference for determining the probabilities and utilities of alternate search paths (Hansson and Mayer, 1989). That is, instead of relying on heuristic estimates for evaluating nodes, Hansson and Mayer examine the control of search as a problem of gathering information and making decisions based on coherent evidential reasoning.

Several investigators have sought recently to apply probability and utility to computer-based planning. In planning, an automated reasoner must determine a sequence of actions to achieve a goal. Planning requires a consideration of the preconditions and effects of applying alternative operators to change states of the world. Dean and Boddy (Dean and Boddy, 1988) described notions of partial computation and flexibility similar to the ideas presented in (Horvitz, 1987c), with a focus on planning. Dean's group later investigated methods for the control of planning, employing techniques similar to the decision-theoretic methods we had developed for controlling computation under uncertain resource constraints. To date, researchers in Dean's group have performed interesting studies of the use of probability and utility in several planning tasks, including problems with temporal reasoning (Kanazawa and Dean, 1989) and path planning (Boddy and Dean, 1989). In related research, Fehling and Breese explored the application of decision theory to the control of a robot planning problem (Fehling and Breese, 1988). In other work, Agogino and colleagues have sought to apply principles of decision theory to select the best computational model to employ in real-time reasoning in the control of milling machinery (Agogino and Ramamurthti, 1989). In related research on rational problem solving in AI, Doyle has described a representation and analysis of a distributed approach to rationality (Doyle, 1988), and Etzioni and Mitchell have analyzed the use of decision analysis for the control of automated learning (Etzioni and Mitchell, 1989).

4.6 Summary

In this chapter, I described the development of EVC formulae for controlling the nature and extent of probabilistic inference in a time-pressured setting. I highlighted the use of partial characterizations of probabilistic inference to reason about the value of continuing to reason about a problem versus that of taking action in the world. After defining the expected value of perfect computation, I developed a formulation of EVC for probabilistic inference in a decision context, focusing on the analysis of two classes of knowledge about future probability distributions over the truth of a proposition of interest. I enumerated an efficient EVC formulation for probabilistic-bounding algorithms, called EVC/BC, and instantiated the formula for the case of a binary decision problem. Finally, I reviewed related research on decision theory in the control of reasoning. In Chapters 6 and 7, we shall use the EVC formalism to study the value of normative metareasoning. First, in Chapter 5, I shall introduce the bounded-conditioning approximation strategy. This probabilistic bounding strategy satisfies several properties of flexibility. In Chapter 6, I shall show how the Protos system can use EVC for inference to control bounded conditioning and other approximate inference strategies. We shall explore the value of continuing to reason with bounded conditioning under time-pressured contexts.

Chapter 5

A Flexible Inference Algorithm

As I described in Chapter 1, several AI investigators have recently structured and assessed large, complex belief networks. The complexity of inference in belief networks is related to the number and complexity of loops in the networks. A singly connected network or *polytree* is a belief network that has no more than one path between any two nodes. There are efficient algorithms for solving singly connected networks, including a distributed algorithm, developed by Kim and Pearl, that has a time complexity that is linear in the number of nodes of the network (Kim and Pearl, 1983). Belief networks with loops or *multiply connected* networks pose more difficult computational challenges. A variety of exact and approximate methods for performing inference with these belief-networks have been developed. Exact methods for reasoning with multiply connected networks typically exploit the special topologies of sparse belief networks. Several of the approximate methods exhibit resource monotonicity and convergence. Readers unfamiliar with belief network inference strategies may wish to review Section A.3 in Appendix A.

I shall introduce a flexible algorithm for probabilistic inference called *bounded conditioning*. Bounded conditioning demonstrates resource monotonicity in its ability to refine the bounds on probabilities in a belief network, and it converges on a queried probability with the allocation of a complete resource fraction. As such, the approach serves as an example of an inference technique for reasoning under the general conditions of uncertain and varying reasoning resources. The method works by decomposing an inference problem into a set of tractable subproblems, each representing a particular truth assignment or context. With bounded conditioning, a reasoner can consider subsets of all subproblems. We compute logical upper and lower bounds on target probabilities by accounting for the contexts that have not yet been examined. The method orders the solution of the subproblems by the ability of each subproblem to tighten the bounds on the probabilities of interest. As we shall see, the value of performing inference with bounded conditioning can be evaluated with the calculation of EVC/BC, described in Section 4.4. I shall introduce bounded conditioning and present its performance on a complex belief network. Then I shall discuss theoretical characterization of the worst-case performance of bounded conditioning and I shall examine the basis for the stereotypical convergence we have observed in the use of the algorithm. In Chapter 6, I shall describe Protos, a reasoning system that combines bounded conditioning with the principles of normative metareasoning described in Chapter 4.

5.1 Method of Conditioning

The *method of conditioning* is an exact algorithm developed by Pearl (Pearl, 1986) for inference in belief networks. The approach provides the basis for bounded conditioning. The method of conditioning operates by transforming a multiply connected network to a set of singly connected network problems. With this method, dependency loops in a belief network are *broken* by a set of nodes called a *loop cutset*, so named because its members are selected such that every loop (a minimal multiply connected subset of the network) is cut by at least one member of the set. After the loop cutset is identified, the method of conditioning requires the consideration of all possible truth assignments of cutset variables. Each combination of truth assignments defines a distinct cutset instance, and defines a singly connected belief-network *subproblem*. Given observed evidence, the probabilities of variables in each instance are solved with an efficient method for computing conditional probabilities with singly connected networks. Finally, in the method of conditioning, the conditional probabilities computed from the singly connected subproblems are combined to calculate a final probability of interest. The combination method sums the probabilities inferred in each subproblem, weighted by the probability of the instance. The method of conditioning is also referred to as the method of *reasoning by assumptions* because of the key notion of disentangling a belief network into a set of simpler network subproblems by generating a set of assumptions about the value of cutset variables.

Figure 5.1 illustrates the fundamental technique used in the method of conditioning and in the bounded-conditioning approximation method. The simple belief-network in this figure defines a dependency loop. The gray node is the single loop cutset variable. We generate distinct inference subproblems by setting this variable to its possible values (in this case values T and F). After instantiating the cutset node for this network, we can solve two simpler problems. For one subproblem, we assume that the cutset variable is set to T. In the other, we assume that the variable is set to F. We then combine the separate answers, making use of knowledge about the probability of each instantiation.

Unlike the simple example displayed in Figure 5.1, loop cutsets in complex networks usually contain several nodes. We shall refer to cutset instances as $c_1 \ldots c_n$ to denote a specific combination of values of loop cutset nodes $C_1 \ldots C_n$. With the method of conditioning, the number of subproblems that must be solved is equal to the product of the number of values of each node in the loop cutset. That is, we must solve $\prod_{i=1}^{n} \mathcal{V}(C_i)$ where C_i is a node in the cutset, and $\mathcal{V}(C_i)$ is the number of possible outcomes for C_i. Figure 5.2 shows a 5-node cutset (darkened nodes) for the ALARM belief network for intensive-care medicine, introduced in Chapter 4. This cutset has 108 instances.

For binary-valued variables, the number of instances that must be solved is $2^{|\text{cutset}|}$. Because this number grows exponentially with the size of the cutset, it is typically useful to expend effort to search for the smallest cutset. Problems, approximations, and empirical testing of ways of identifying good cutsets in complex networks are discussed in (Suermondt and Cooper, 1988). Techniques for controlling the search for

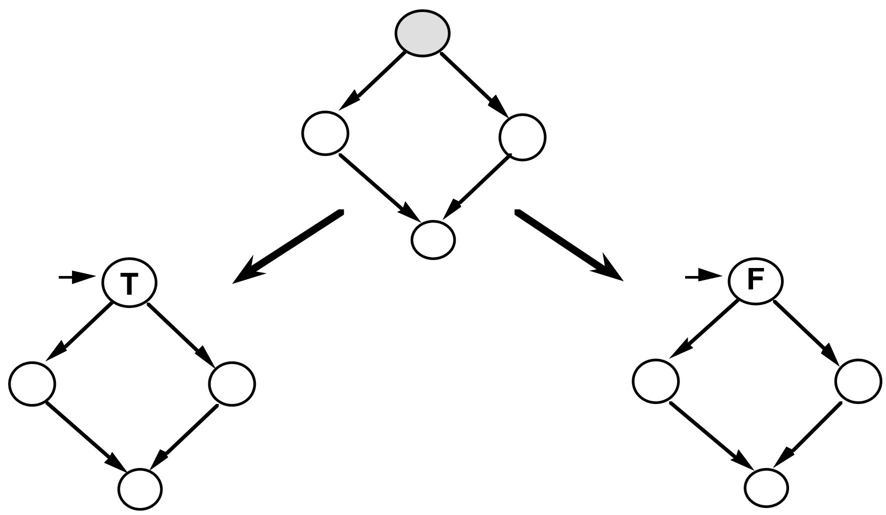


Figure 5.1: Instantiating belief-network loops with a cutset.
Setting variables in a belief-network loop to particular values breaks the loop. In this case, we generate two instances by setting the binary-valued cutset node (gray node) to T (true) and F (false). With the method of conditioning, a set of nodes that can be instantiated to render the network singly connected are identified. This cutset determines the number and configuration of subproblems that must be solved.

smaller cutsets are addressed in (Breese and Horvitz, 1990).

The method of conditioning relies on the computation of the joint probabilities or *weights* of the loop-cutset variables. These probabilities represent the likelihood of the state of affairs assumed in each subproblem. We compute the prior joint probabilities of the loop-cutset variables, given background information ξ, $p(c_1 \ldots c_n|\xi)$, during a preparatory phase referred to as the initialization of the belief network. At run-time, we compute the posterior weights, $p(c_1 \ldots c_n|E, \xi)$ for new observations E.

An algorithm for computing the prior joint probabilities from information in the belief network is described in Suermondt and Cooper, 1989. Initially,

$$p(c_1 \ldots c_n|\xi) = p(c_1|\xi)p(c_2|c_1, \xi) \ldots p(c_n|c_1 \ldots c_n, \xi)$$

During computation for initializing the network, we calculate, for each cutset instance, the marginal probabilities for each node in the network, given the values assigned to the loop-cutset nodes in that instance. For each value x of node X, and for each

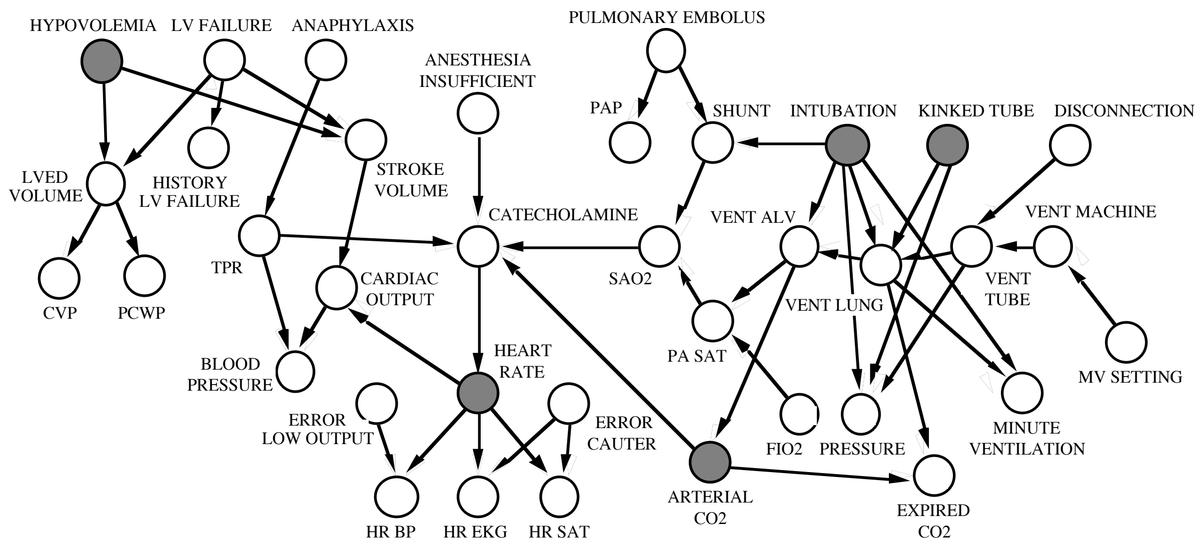


Figure 5.2: A cutset for the multiply connected belief network for ICU diagnosis. The darkened nodes are a cutset for the multiply connected belief network introduced in Figure 4.4 in Chapter 4.

instance $c_1 \ldots c_n$, we obtain

$$p(x|c_1 \ldots c_n, \xi)$$

Thus, calculating $p(x|\xi)$,

$$p(x|\xi) = \sum_{c_1 \ldots c_n} p(x|c_1 \ldots c_n, \xi) p(c_1 \ldots c_n|\xi)$$

If we change the truth status of one or more propositions in a belief network, as is the case when we observe evidence, we must update the weights of each instance subproblem. We update the probabilities for each loop-cutset instance such that, when added together, they are equal to the joint probability of the loop-cutset nodes and the evidence. We perform this update for each instance subproblem by multiplying the prior probability of each instance by the probability of the evidence given the truth assignments that define the instance (Pearl, 1986). If we observe a value e of node E, we calculate the new probability of an instance as follows:

$$p(c_1 \ldots c_n|e, \xi) = \alpha p(c_1 \ldots c_n, e|\xi) = \alpha p(e|c_1 \ldots c_n, \xi) \times p(c_1 \ldots c_n|\xi)$$

where

$$\alpha = \frac{1}{p(e|\xi)} = \frac{1}{\sum_{c_1 \ldots c_n} p(e|c_1 \ldots c_n, \xi) p(c_1 \ldots c_n|\xi)}$$

After the revised probabilities of each instance are calculated, we can apply one of several probabilistic inference algorithms to propagate evidence in the polytree subproblems to solve for the posterior probabilities of all nodes in each polytree instance. For example, we can use Pearl's distributed algorithm for propagating evidence in singly connected belief networks (Pearl, 1986). Thus, we assign a probability to each value x of node X, $p(x|e, c_1 \ldots c_n, \xi)$ This, in turn, allows us, at any time, to obtain $p(x|E, \xi)$ for any node; we sum the belief in x for all instances, weighted by the likelihood of each instance:

$$p(x|e, \xi) = \sum_{c_1 \ldots c_n} p(x|e, c_1 \ldots c_n, \xi) p(c_1 \ldots c_n|e, \xi)$$

For additional evidence, we repeat this procedure, each time multiplying the probability assigned to an instance by the probability of the observed value given that instance, and renormalizing the weights. Thus, the method of conditioning provides a mechanism for performing general probabilistic inference in multiply connected belief networks.

5.2 Bounded Conditioning

The computational complexity of the method of conditioning is an exponential function of the size the cutset. If we have a large number of cutset instances, exact probabilistic inference using this method may not be feasible in situations where sufficient time is not available or where delay is costly. To provide computation that has maximal value to a computational decision system or system user under varying and uncertain deadlines and urgencies, we must balance the net costs and benefits of computation. As opposed to using inference techniques, such as the method of conditioning, that completely solve a problem in some specified period of time, we may wish to do inference in a manner that provides results that span a range of precision. As described in Section 2.3.1, flexible approaches to inference provide us with an opportunity to choose an optimal halting time, given the costs and benefits of computation. These strategies allow us to distill the most inferential value out of a problem under an uncertain deadline by continuing to refine a result until the deadline arrives. I shall now introduce a probability-bounding method, based on the method of conditioning, that allows us to perform inference in situations where we may not have the luxury of a complete analysis.

With *bounded conditioning*, we do not necessarily consider all problem instances. Rather, we consider instances based on truth assignments in order of the probability of the truth assignments. Rather than being constrained to wait until a point probability is generated, we can determine, with a fraction of the complete resources, upper and lower bounds on a target probability—the probability that would be calculated with sufficient computation to solve the problem. We obtain exact upper and lower bounds on the probability of each value of each node in the network by computing the maximal positive and negative contributions of the yet unsolved problem instances. We continually probe the unexplored portion of the reasoning problem to order the analysis of instance subproblems by their expected contribution on the tightening of bounds.

5.2.1 Inference from a Complete State

In bounded conditioning, we initialize a belief network as we do in the exact method of conditioning. After observing some evidence, we (1) calculate the revised weights of each subproblem, (2) sort the subproblems by weight, (3) update each instance in sequence, and (4) integrate the results of each loop-cutset instance, based on the weights of the instances and information about the contribution of the unexplored problem instances. We shall now examine calculi for computing the upper and lower bounds of a probability of interest in cases where we begin inference on a completely initialized belief network and in cases where we observe new evidence before we complete inference with previous observations.

Let us first consider the case where a fully-initialized belief network is updated given the observation of a piece of evidence. After observing value e of evidence node E, we first recalculate the loop-cutset weights for each instance in the context of the evidence. Next, we solve the marginal probabilities of values of nodes in each instance, in order of the prior weight of the instances. To simplify our notation, we shall associate with the ith instance $c_1 \ldots c_n$ of the cutset an integer label i, and designate $p(c_1 \ldots c_n|\xi)$ as the weight w_i of that instance. We shall use w_i^* to distinguish the weight of an instance given the most recent observation e', $p(c_1 \ldots c_n|e', e, \xi)$, from the previously calculated weight of an instance, based on the initial state of the network or on a previous observation e, $p(c_1 \ldots c_n|e, \xi)$.

For those loop-cutset instances that we have updated, we compute $p(x|e, \text{instance } i, \xi)$. For the loop-cutset instances that we have not yet updated, we know with certainty that

$$0 \leq p(x|e,\ \text{instance } j, \xi) \leq 1$$

Therefore, for any node X and value x, we can obtain a lower bound on $p(x|e, \xi)$ by substituting 0 for those probabilities we have not yet calculated. We can calculate an upper bound on $p(x|e, \xi)$ by substituting 1 for $p(x|e, \text{instance } j, \xi)$.

Let us assume that we only propagate the evidence through the network for a subset of instances 1 through j; therefore, we do not update the probabilities for instances $j + 1$ through n. After propagating the evidence for instances 1 through j, we can calculate bounds on $p(x|e, \xi)$ as follows:

$$\begin{aligned}
\text{Lower bound on } p(x|e, \xi) &= \sum_{i=1}^{j} p(x|e, \text{instance } i, \xi) \times w_i^* + \sum_{i=j+1}^{n} 0 * w_i^* \\
&= \sum_{i=1}^{j} p(x|e, \text{instance } i, \xi) \times w_i^*
\end{aligned}$$

Similarly, for the upper bound,

$$\begin{aligned}
\text{Upper bound on } p(x|e, \xi) &= \sum_{i=1}^{j} p(x|e, \text{instance } i, \xi) \times w_i^* + \sum_{i=j+1}^{n} 1 \times w_i^* \\
&= \sum_{i=1}^{j} p(x|e, \text{instance } i, \xi) \times w_i^* + \sum_{i=j+1}^{n} w_i^*
\end{aligned}$$

Thus, the difference between these bounds is

$$\text{Upper bound} - \text{Lower bound} = \sum_{i=j+1}^{n} w_i^*$$

Note that the width of this interval is the same for every marginal probability in the network; after j instances have been updated, our uncertainty about the probability of a value of a node in the network does not depend on the node or the value.

This form of bounded conditioning assumes that we begin with a completely initialized network. We are in this state after initialization of the network, and after we solve all instances of a problem given evidence. Such bounding from a complete state is most useful in situations where pieces of evidence are seen at intervals long enough to allow the eventual complete updating of the network, yet where decisions may have to be made as soon as possible after the observation of that evidence.

5.2.2 Bounding from an Incomplete State

We shall now generalize the bounding calculus to allow us to update a network with new evidence before previous evidence has been completely analyzed. Recall that the revised weight for an instance, in light of new evidence, is obtained by multiplying the old weight for that instance by the probability of the observed evidence in that instance. Then, this product is normalized by dividing it by the marginal probability of the evidence. To compute the weights, we must first calculate the marginal probabilities within each instance. If we did not update the belief in values of the nodes in a particular instance when we added the last piece of evidence, it is not possible to obtain the probability of the new evidence, since all of the instances have not been updated. To reason about the relevance of additional pieces of evidence, given a previously incomplete analysis of a subset of instance subproblems requires us to apply a bounding analysis to the weights themselves. This makes our bounding calculus more complicated.

Suppose we observe evidence e for node E, recalculate the weights for all instances given this evidence, and then only propagate this evidence for a subset of instances $i = 1$ to j. After completing the propagation of evidence in instance j, we observe value f for node F. To update probabilities of interest given e and f, we need to compute revised weights $w_i^* = p(\text{instance } i|e, f, \xi)$. For $i = 1$ to j we can calculate the belief in the conjunction of the new evidence and the old. For $i = 1$ to j we know

$$p(f, \text{instance } i|e, \xi) = p(f|e, \text{instance } i, \xi) \times p(\text{instance } i|e, \xi)$$

If we knew this for all instances i, we would simply normalize; this is equivalent to

$$w_i^* = p(\text{instance } i|e, f, \xi) = \frac{p(f, \text{instance } i|e, \xi)}{\sum_k p(f, \text{instance } k|e, \xi)}$$

However, we cannot normalize $p(f, \text{instance } i|e, \xi)$ over all instances i, as we do when we begin with a complete set of instance weights. Now, in addition to bounding the target probabilities through considering the minimal and maximal contributions of unsolved instances, we need to consider upper and lower bounds on the weights themselves.

For $i = j + 1$ to n, we know only that

$$0 \leq p(f, \text{instance } i|e) \leq p(\text{instance } i|e, \xi)$$

The upper bound follows from the fact that $p(a \wedge b) \leq p(b)$. Therefore, we can use our last-calculated cutset weight for instance i as an upper bound on the weight for that instance. For instances $i = j + 1$ to n, we know that

$$0 \leq p(\text{instance } i|e, f, \xi) \leq 1$$

Thus, $0 \leq w_i^* \leq 1$ for $i = j + 1$ to n.

Therefore, we can bound instances $i = 1$ through j,

$$\frac{p(f, \text{instance } i|e, \xi)}{\sum_{k=1}^{j} p(f, \text{instance } k|e, \xi) + \sum_{k=j+1}^{n} p(\text{instance } k|e, \xi)} \leq w_i^* \leq \frac{p(f, \text{instance } i|e, \xi)}{\sum_{k=1}^{j} p(f, \text{instance } k|e, \xi)} \tag{5.1}$$

Now we can return to updating the cutset instances. Let us assume that for the new observation f, we only propagate evidence in instances 1 through h where $h \leq j$. Because instances $j + 1$ through n were not updated when the last piece of evidence was added, they cannot be updated now either. Thus, we can update and sort only instances 1 through h. Let $P^L(x|e, f, \xi)$ and $P^U(x|e, f, \xi)$ represent, respectively, the upper and lower bounds on the conditional probability of interest, w_i^{L} indicate a lower bound on the weight of instance i (left side of Equation 5.1), and w_i^{U} indicate an upper bound on the weight (right side of equation 5.1). After updating instances 1 through h with the new evidence, we have the following bounds on $p(x|e, f, \xi)$:

$$P^L(x|e, f, \xi) \quad = \quad \sum_{i=1}^{h} p(x|e, f, \text{instance } i, \xi) \times w_i^{\mathrm{L}}$$

$$P^U(x|e,f,\xi) = \sum_{i=1}^{h} p(x|e,f,\text{instance } i,\xi) \times w_i^{\mathrm{U}} + \sum_{i=h+1}^{j} w_i^{\mathrm{U}} + \sum_{i=j+1}^{n} w_i$$

5.3 Performance of Bounded Conditioning

The performance of bounded conditioning has been analyzed with several belief networks. Let us examine sample runs with the ICU network described in Section 4.2. The behavior of bounded conditioning was studied for several loop cutsets for this network. One cutset consists of 5 nodes that lead to 144 different singly-connected network subproblems. The convergence of the bounds with the solution of additional subproblems, for a sample update, is displayed in Figure 5.3. The figure shows the upper and lower bounds, as well as the mean probability as the belief-network subproblems are sequentially solved. Figure 5.4 shows the decay of the interval between the upper and lower bound on a probability of interest with computation. We found that the bounds interval for many updates can be approximately modeled with a negative exponential, $int = e^{-kt}$. In the case portrayed in Figure5.4, convergence is modeled with a decay constant of $k = 0.02$. Figure 5.5 shows the convergence of bounded conditioning for the same problem, making use of a different cutset.

Figure 5.6 shows the behavior of the iterative application of bounded conditioning to incomplete states, as new evidence is observed before earlier evidence is completely analyzed. The graph shows that the interval of the bounds at maximal convergence grows with each new observation.

This convergence information can be used to calculate an EVC of inference, enabling us analyze the expected value of continuing to apply the bounding algorithm in the context of the costs and benefits of taking action in the world. In fact, the convergence information, based on the incremental reduction of the bounds interval by the weight of each subproblem, is just the *belief-constellation* knowledge, $p(p_t(\phi))$, required by the EVC/BC calculation described in Section 4.4. Recall that the EVC/BC makes use of future intervals between the lower and upper bounds on a probability of interest with computation. This knowledge is made available after the initialization

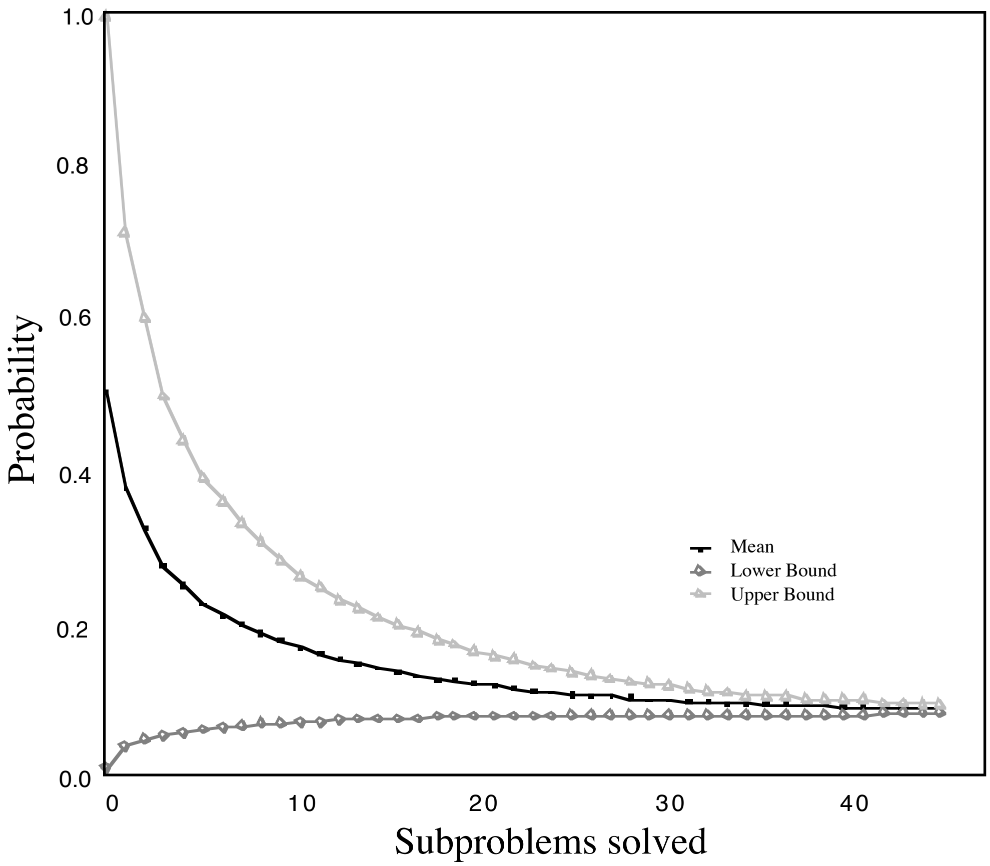


Figure 5.3: Convergence of bounded conditioning.
This graph shows the stereotypical convergence of upper and lower bounds on a probability of interest generated by bounded conditioning, in response to the observation of evidence. Each instance subproblem requires approximately 5 seconds of computation.

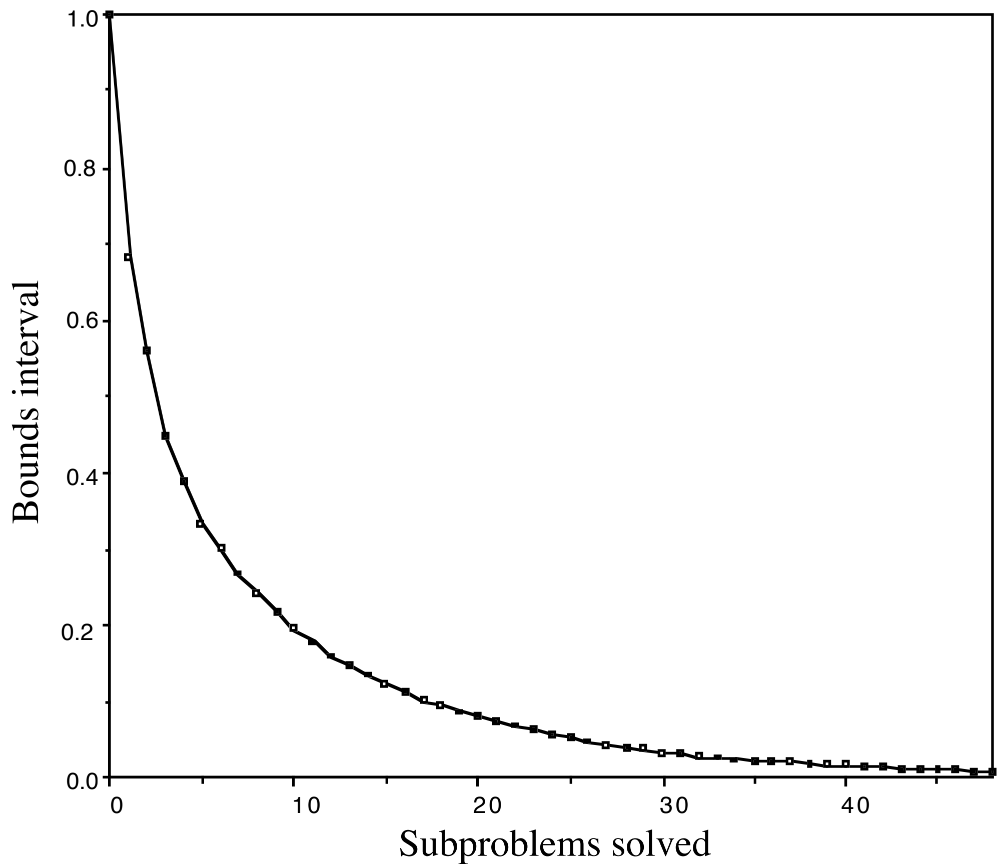


Figure 5.4: Convergence of the bounds interval.
The convergence of the bounds interval (upper bound - lower bound) on a probability with computation can be modeled by an exponential-decay model.

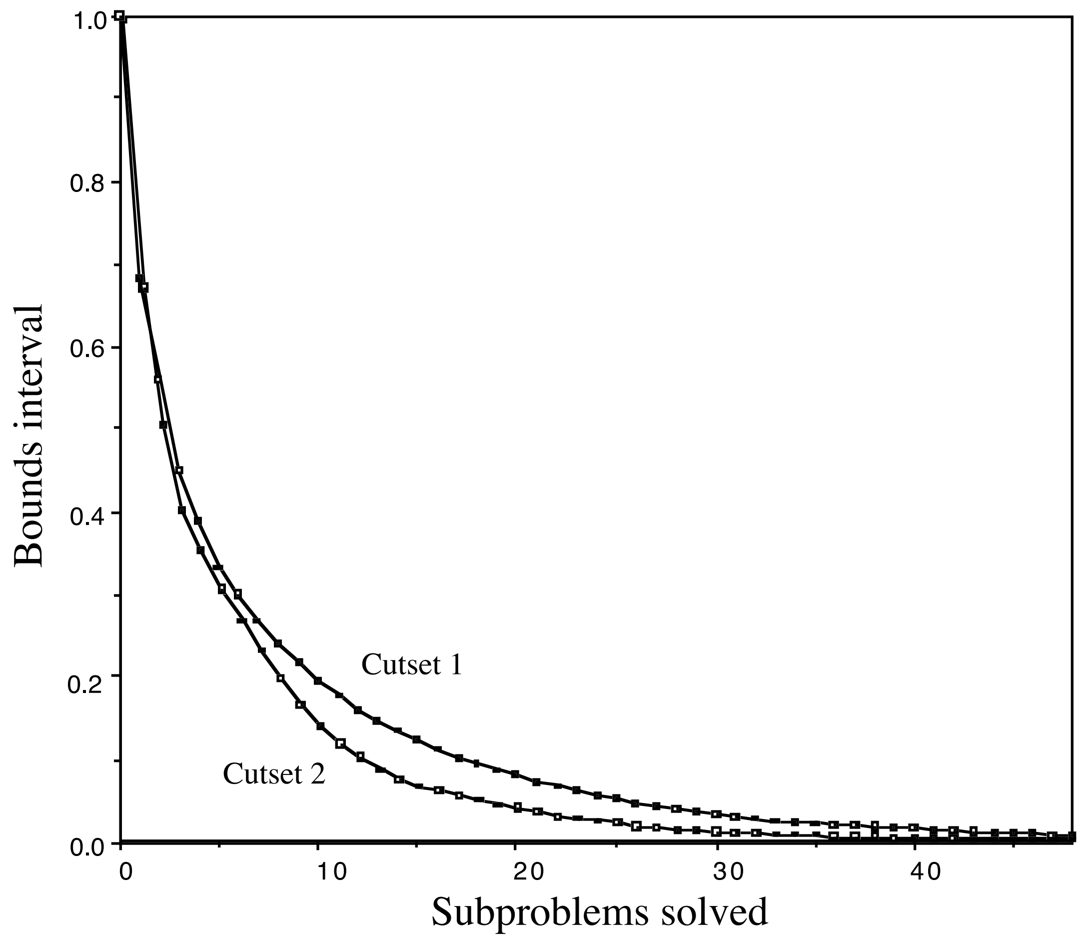


Figure 5.5: Convergence for a different cutset.

This graph shows the convergence behavior of bounded conditioning using two different cutsets to decompose the multiply connected belief network on the same inference problem.

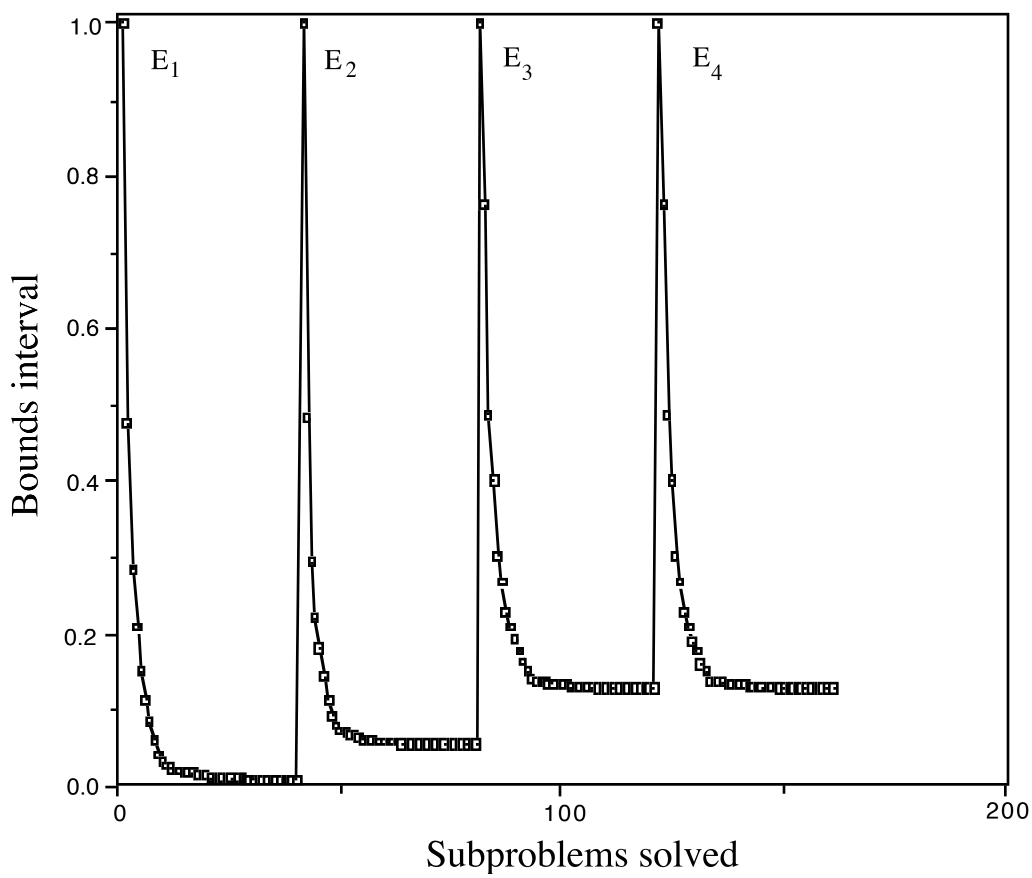


Figure 5.6: Updating from an incomplete state.

When new evidence is observed before the complete solution of an inference problem, we must bound the weight instances. This graph shows the convergence of the bounds interval in response to a sequence of 4 sequential observations.

of a set of instance weights. We shall review the use of the EVC/BC with bounded conditioning in Chapter 6.

5.4 Caching of Instance Weights

An approach to extending bounded conditioning effectively to the case of multiple observations, without resorting to the technique of bounding the weights of instances is to compute instance weights ahead of time and to store them for retrieval at run-time. Alternatively, we can compute and cache a small subset of potentially useful instance weights during the idle time between observations.

5.4.1 Offline Caching

The offline compilation of evidence weights is in the spirit of recent work on the compilation of probabilities by Herskovits and Cooper (Herskovits and Cooper, 1989). These investigators explored methods for selecting important probabilities for offline caching. The caching of all probabilities that can be inferred from a belief network of binary variables would require the storage of 2^n numbers, where n is the number of nodes in the belief network. If we consider a subset of belief network nodes to serve as a stable set of observable evidence E, the compilation of of instance weights requires the storage of a set of weights for each possible combination of evidence; we have to store $2^{|cutset|} \times 2^{|E|}$ numbers. Thus, the quantity of memory required to cache instance weights grows exponentially with the number of pieces of evidence included in the cache and exponentially with the size of the cutset. The use of bounded conditioning reduces the memory needed to store compiled probabilities by a factor of $2^{n-|E|-|cutset|}$. Large networks with small cutsets are candidates for the use of compilation in conjunction with real-time bounded conditioning.

In a diagnostic setting, decision makers typically are not interested in the complete joint probability space represented by a belief network. Instead, they interested in the probabilities of special hypothesis nodes H (e.g., diseases), given observed evidence. If we have a stable set of binary-valued hypothesis and evidence nodes, we need only to cache $|H| \times 2^{|E|}$ probabilities. In such cases, the relative savings of the use of bounded conditioning depends on the number of loop cutset nodes versus the number of special hypotheses, the cost of memory, and the cost of the delay associated with the use of bounded conditioning.

We can reduce the quantity of memory used for compiling instance weights for bounded conditioning by (1) caching loop-cutset instance weights for sets of observations that are associated with time-critical decisions, (2) weakening the convergence criteria of bounded conditioning, so that the algorithm halts at a predetermined bounds interval, or (3) adjusting such predetermined final bounds intervals to reflect the criticality of sets of observations. The effectiveness of (2) and (3) is highlighted by the rapid convergence of the bounds on a probability with a small fraction of instance weights. All of these approaches could reduce dramatically the number of probabilities we need to store for the effective general application of bounded conditioning.

5.4.2 Idletime Caching

We wish to make the best use of the idle time between the completion of a previous update and the observation of new evidence. In many applications, there may be time between observations for planning future updating. In domains where we expect infrequent bursts of sequences of observations, it can be effective to compute instance weights for anticipated future observations during the idletime between observations. The computation of instance weights can be directed by the likelihood of new observations, conditioned on the current state of the world.

5.5 Multiple Approximation Methods

There is potential for applying several different approximation methods in conjunction with bounded conditioning, with the hope that the differences in the nature of the approximations can be used to produce a result that captures the best aspects of each algorithm.

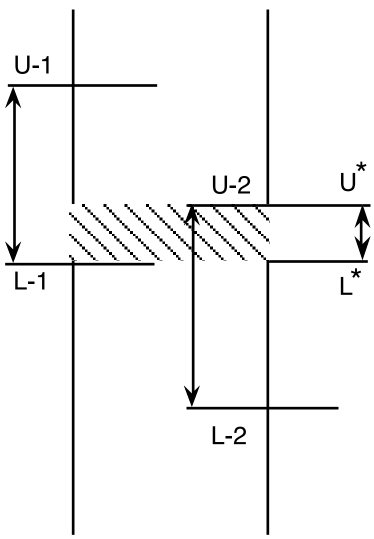
5.5.1 Bounded Conditioning for Topological Editing

We can use bounded conditioning to break a subset of loops in a multiply connected network, producing alternate mixtures of singly connected and multiply connected subproblem instances. The selective assignment of states of truth and use of bounded conditioning can provide an effective means of *topologically editing* a network to generate subproblems that can be solved efficiently by other algorithms. For example, we can break a complex problem into a set of multiply connected problems that can be solved effectively by simulation or by the clique-tree methodology such as the method of Lauritzen and Spiegelhalter (Lauritzen and Spiegelhalter, 1988). More generally, the controlled decomposition of a belief network by bounded conditioning can generate subproblems that can be solved with different methods depending on the topology of the subproblem.

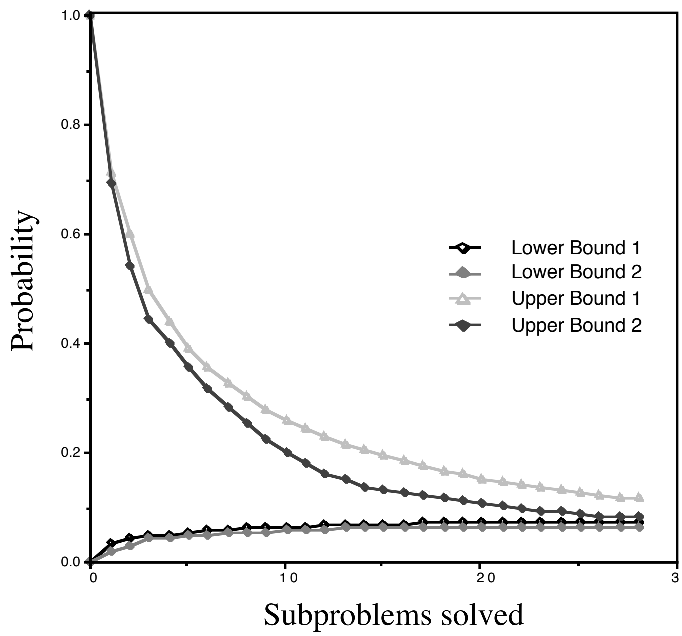
5.5.2 Concurrent Bounded Conditioning

There are typically several loop cutsets for a belief network. Alternative cutsets update the upper and lower bounds of a probability of interest in a different manner. In preliminary experimentation, I investigated gains that could be ascertained through the concurrent processing of a belief network with two different bounded-conditioning analyses. As indicated in Figure 5.7(a), given a set of two or more bounding analyses, we select the highest lower bound and the lowest upper bound to construct final upper and lower bounds on a probability of interest. The less the overlap among different sets of bounds, the greater the benefits of concurrency. Figure 5.7(b) displays graphs of analyses based on two different loop cutsets. The convergence of the final bounds is just the interval between the greatest lower bound and the smallest upper bound. In analyses to date, the overhead of executing two different problems has been greater than the gains. There may exist cases, however, where concurrent bounding yields a net gain. However, we may find that the gains in convergence may outweigh the overhead. In related work, a promising parallel implementation of bounded conditioning that updates problem instances concurrently has been implemented.[1]

[1] The parallel version of bounded conditioning was developed by Adam Galper.



(a)



(b)

Figure 5.7: Concurrent application of two different cutsets.
(a) The goal of concurrent bounded conditioning is to solve simultaneously different bounding problems, and to combine the different bounds (U-1, L-1 and U-2, L-2) to construct final bounds (U^*, L^*) on a probability of interest. (b) Upper and lower bounds associated with two concurrent bounded-conditioning analyses of an ICU inference problem.

5.5.3 Simulation for Calculating Instance Weights

There can be great synergy in the application of alternative algorithms for performing inference about components of the bounded-conditioning problem. For example, there is promise in integrating stochastic-simulation methods (Shachter and Peot, 1989; Chavez and Cooper, 1989b) for computing cutset weights conditioned on sets of evidence. A more robust version of bounded conditioning for multiple pieces of evidence may result from this work.

5.6 Theoretical Analysis of Convergence

The rapid convergence of bounded conditioning provoked me to investigate grounds for the method's behavior. Insight about the convergence of bounded conditioning can be gained by considering the distribution of instance weights. My analysis has focused on the *asymmetry* in the way probability is apportioned to values of cutset nodes.

5.6.1 Worst-Case Convergence

Consider a belief network that is cut by a loop cutset of n binary nodes. Let us consider how probability is apportioned to values of nodes in the cutset of a belief network. Recall that the instances are ordered by their associated weights. The sorting operation insures that an instance in any position of the ordered list is greater than or equal to the weight of any successor in the list. The weight of the first, or largest instance, takes on a minimal value when it is equal to the weight of the smallest instance in the sorted list. This condition is satisfied only when all of the weights are assigned an equal value. Thus, in the worst case, all subproblem instances have the same weight. With n binary nodes, we have 2^n instances, each with weight 2^{-n}. In the worst case, bounds converge with 2^{-n} with the solution of each problem instance. After expending resource to recompute the weights, the worst-case convergence of the bounds interval at time t is

$$1 - (2^{-n} \times \frac{t}{k})$$

where k is the amount of time required by each instance for solution.

Figure 5.8 shows an example of worst-case linear convergence for a loop cutset of 15 nodes. This linear convergence is the slowest rate of refinement we can expect with bounded conditioning. Even in such worst cases, the utility structure of the decision problem—for which the inference is being performed—can dictate that we need to solve only a portion of the entire inference problem to derive a great fraction of the value of perfect inference.

5.6.2 Better-Case Convergence

Asymmetric distributions over the conditional probabilities of alternative values of specific cutset nodes, given values assumed for other nodes in a cutset, allow for a wide range of differences among the weights for instances. Sorting and sequentially solving these subproblems enables a reasoner to take advantage of the nonlinearity in weights with subproblems. Such situations often enable a reasoner to capitalize on a disproportionate amount convergence for early computation.

Consider the case where we again have a loop cutset of n binary nodes. Now, however, we have an identical asymmetric contribution for values of each node in the cutset, within each instance. Each node takes on one value (e.g., "true") with probability p, and another value (e.g, "false") with probability $1-p$. Within such a network, we have several sets of instances with equal weight. We are assuming that the dependence among cutset nodes is insignificant to this analysis. In particular, we have sets of instances with weight

$$p^{n-j}(1-p)^j$$

each of cardinality $\frac{n!}{j!(n-j)!}$, from the largest to smallest weights as j varies from 0 to n. The bounds interval, based on an incomplete analysis, in this case is described by

$$1-\sum_{j=0}^{m}\frac{n!}{j!(n-j)!}p^{n-j}(1-p)^j$$

where m refers to the index of the set of instances of smallest weight $[p^{n-m}(1-p)^m]$ considered in the incomplete analysis.

In juxtaposition to a scenario of worst-case convergence, Figure 5.8 displays a better-case convergence, where each of the 15 loop-cutset nodes takes on the value true with probability 0.75, and the value false with probability 0.25. This graph shows a piecewise-linear convergence at different rates for each value of j. The rate of convergence with the solution of subproblems is maximal at the outset of inference. The proportions of total problem instances analyzed ($2^{15} = 32,768$) are listed on the x axis. This analysis does not include the possible effects of dependencies among cutset nodes that may exist in a real belief network. However, our analysis for the case of the homogeneous loop cutset can give us intuition about the convergence of bounded conditioning on the distributions of instance weights generated by real belief networks.

5.6.3 Increase in Problem Difficulty with Cutset Size

We can explore the implications of a homogeneous asymmetric distribution over the values of cutset nodes for the time required by bounded conditioning as the number of cutset nodes grows. As described in Section 5.1, we know that the cost of complete inference with bounded conditioning grows exponentially with the size of the cutset. However, we are not interested solely in the computational complexity of inferring a point probability. We wish to determine also how the resources required for solving various *fractions* of the problem changes with the growth of the number of cutset nodes.

Let us assume that the probabilities of the values of binary cutset nodes are 0.8 and 0.2. We can use Equation 5.6.2 to calculate the number of problem instances that would have to be solved to tighten the bounds on a probability of interest to any specified interval. Such an analysis reveals great disparity in the time required for tightening the bounds and for converging on a point probability. Figure 5.9 shows the difference in the growth of difficulty for computing a point probability and the difficulty of computing a bounds interval of 0.5. For fourteen nodes, solution of the complete problem requires the computation of 2^{14} or 16,384 loop-cutset instances. However, computing bounds of 0.5 requires only the solution of 107 subproblems—a solution of less than 0.007 of the total problem. Problem growth curves for bounds

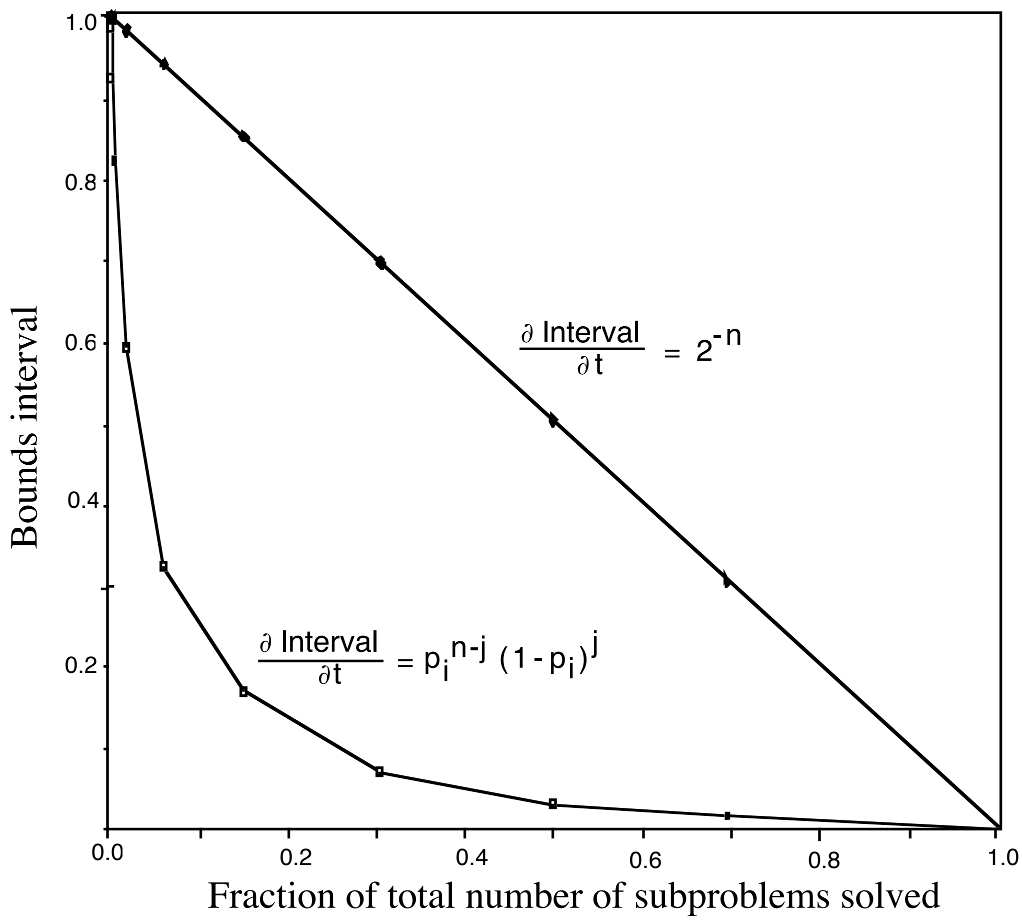


Figure 5.8: Two theoretical scenarios.

The linear, worst-case (upper curve) and better-case (lower curve) convergence of bounds for a marginally independent loop cutset consisting of 15 nodes. The better-case convergence is based on an assumption of homogeneous asymmetry in the distribution over the probability of values for each node $(0.75, 0.25)$.

intervals between 0.5 and 0.0 lay at increasingly higher positions between these two curves.

The plots in Figure 5.10 graphically demonstrate with a linear and exponential decay model, how the growth in the difficulty of generating a particular bounds interval grows significantly more slowly with increases in the size of an inference problem than it does with the growth of the complete solution of the problem. Figure 5.10(a), displays the traditional worst-case expected growth of difficulty for solving all portions of a problem (similar to the worst-case plot at the top of Figure 5.8). In this case, the time required to compute a portion of a desired attribute, such as one-half of the original bounds ($t_{.5}$) grows linearly with the growth of the whole problem. In contrast, as highlighted in Figure 5.10(b), solutions like bounded conditioning tend to deliver a rate of refinement on a result that is inversely proportional to the portion of the problem that has been solved (similar to the better-case plot at the bottom of Figure 5.8). For such problems, we can expect a reduced growth rate for solving portions of the problem.

The bounded conditioning example suggests that we should be sensitive to attributes of partial results that are not as affected by increases in problem size as is the solution of the complete problem. Indeed, we may be able to identify desired attributes, such as the bounds on a probability of interest, that require a quantity of time that grows at a rate described by a low-order polynomial with increases in the size of the problem instance—even when the amount of resource required to solve the entire problem grows exponentially with increases in the problem size.

5.7 Summary

I have described the bounded conditioning method for probabilistic inference and explained how we can use the strategy to compute bounds on the marginal probabilities for any node in a belief network. Bounded conditioning exhibits the useful properties of continuity, monotonicity, and convergence, enabling a reasoner to exchange arbitrary quantities of computational resource for incremental convergence on probability bounds. The approach, and its future extensions, promise to be useful in

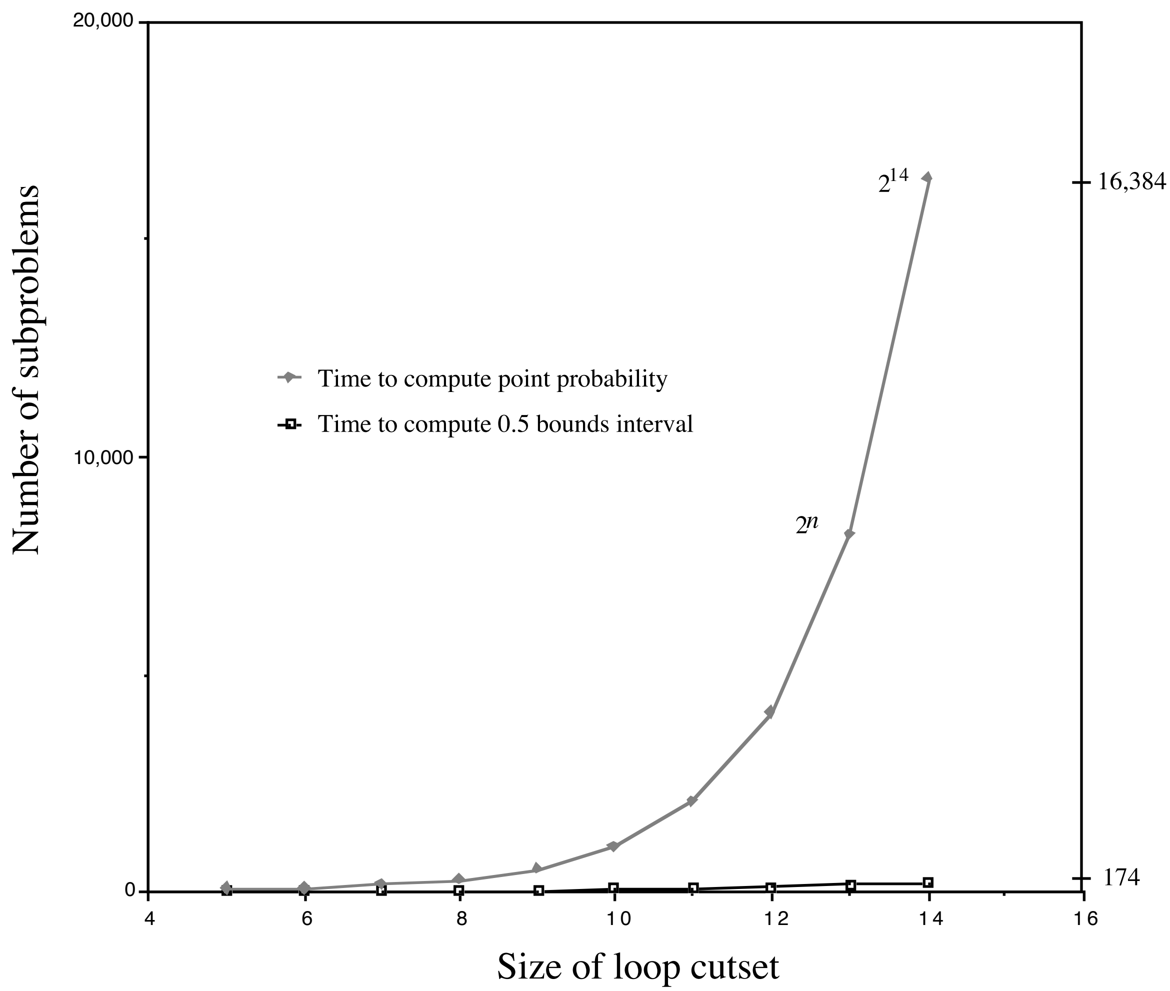


Figure 5.9: Disparity in the growth of difficulty with problem size.
A graph demonstrating the great difference in the complexity of solving for bounds of width 0.5 and computing until reaching a point probability with a model of bounded conditioning based on a homogeneous assignment of 0.8 and 0.2 to binary valued cutset nodes.

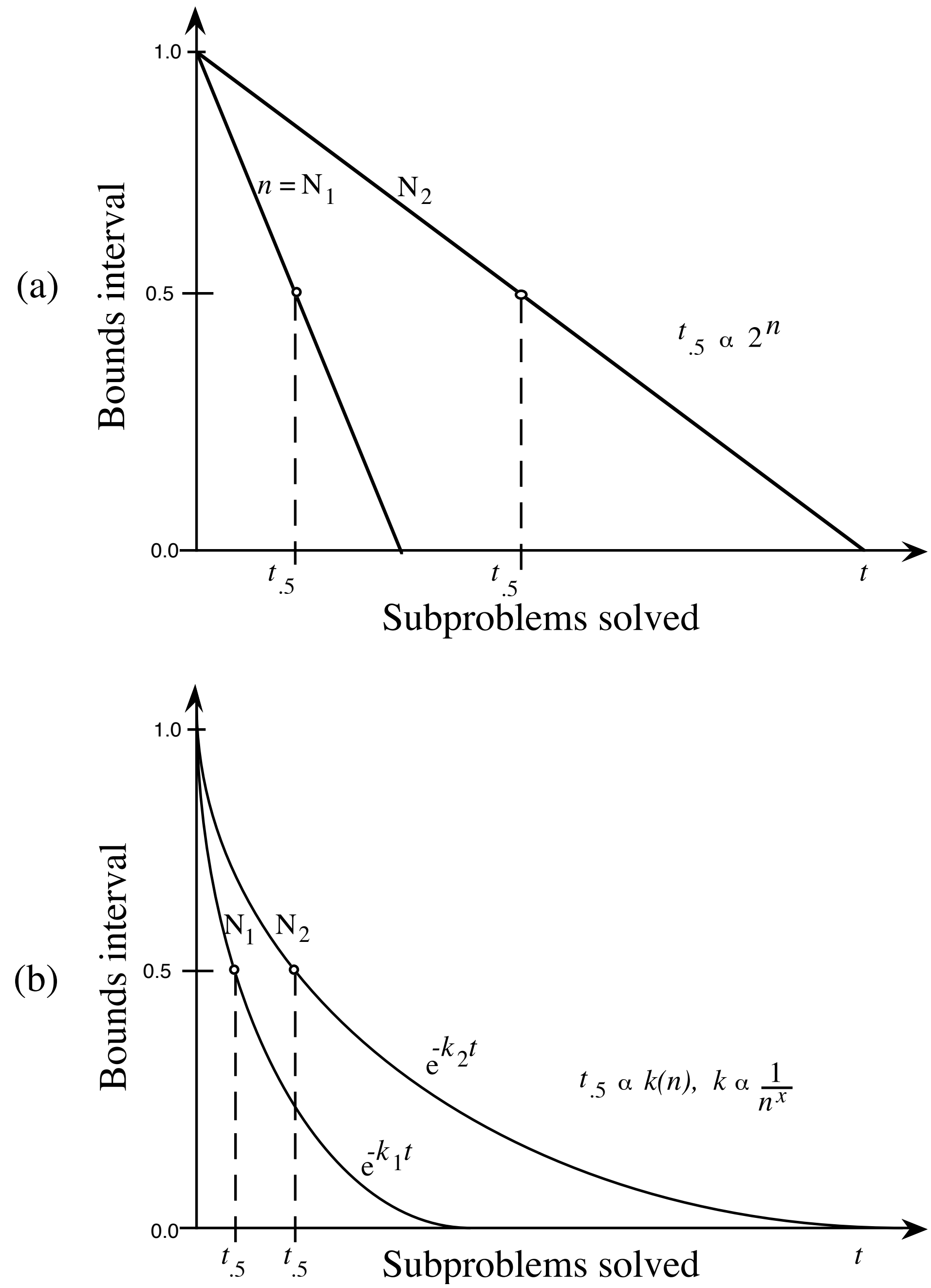


Figure 5.10: Basis for disparity in growth rates.
(a) This graph demonstrates how, in the worst case, the time needed to solve a portion of a bounded-conditioning problem can grow as fast as the time needed to solve the entire problem. (b) Bounded conditioning takes advantage of asymmetries in instance weights. This figure shows how a property of an algorithm, such as the "halflife" on bounds ($t_{.5}$), might grow only as a power of the size of a problem, while the complexity of solving the complete problem grows exponentially. The graph captures intuitively the basis for the computed results displayed in Figure 5.9.

reasoning under the general conditions of uncertainty in available reasoning resources. We have found, in complex belief networks, that the method solves a great portion of a probabilistic inference problem with a solution of a fraction of the total number of instance subproblems. Future efforts will seek to characterize the method more fully and to take advantage of the structure of particular belief networks. Areas of opportunity for additional research include mixing compilation of weights with deliberative concurrent bounded conditioning with multiple cutsets, idletime reasoning about expected updating, the compilation and caching of instance weights, and the use of complementary approximation methods for updating the weights on instance subproblems. We shall turn, in Chapter 6, to this use of bounding algorithms in normative metareasoning about beliefs and actions.

Chapter 6

Protos: Implementation of a Reflective Decision System

I shall describe, in this chapter, the architecture and functionality of Protos, a prototype metareasoning system that demonstrates key principles of bounded rationality. The system exercises ideal control of probabilistic inference under resource constraints, considering the stakes of a decision at hand and the time-dependent utilities of outcomes. Protos was designed to demonstrate the control of decision-theoretic inference for applications that use large probabilistic models to make decisions in time-critical contexts.

Protos dynamically adjusts the length of time it dwells on an inference problem before acting, depending on several classes of knowledge. The system considers the probability distributions most recently computed and the expected refinement of those probability distributions with additional computation time. Protos analyzes such information about current belief and future belief, in conjunction with information about the change in utility of alternative outcomes with delay, to make decisions about the value of continuing to reflect, versus that of taking action. Protos analyses have demonstrated that the optimal quantity of decision-theoretic inference to perform can

Figure 6.1: A screen from Protos justifying the system's reasoning and metareasoning. When Protos is in an interactive mode, it produces a set of graphs to justify its decisions about computation and action. The top row of graphs includes (from left to right) a graph displaying the refinement of a probability distribution, information used to predict the convergence of a probability distribution, and the EVC of an inference strategy. The bottom row includes (from left to right) a graph showing the time-dependent utilities of action and a graphical utility analysis of a recommended action.

depend greatly on the degree of belief in states of the world, outcome utilities, and time-dependent costs of delay.

I developed an interactive mode for justifying Protos' control decisions. Although the graphical interface was developed for validating and illustrating the rationale behind Protos' decisions, such justification abilities may be required in real-world applications of the principles of bounded rationality. Future real-world versions of reflective reasoners like Protos, even in autonomous decision-making situations, may need to justify their behaviors with expressive human-oriented interfaces and explanation facilities. An ability to justify decisions under bounded resources clearly may be especially important in systems that are granted the responsibility for guiding computation in high-stakes decision making.

When an investigator invokes the interactive mode, Protos presents graphically a summary of its reasoning and metareasoning. The main screen of Protos is displayed in Figure 6.1. Protos displays several classes of information that are crucial in the ideal control of probabilistic inference. The system can display a comprehensive trajectory of inference extending from the state of belief at the initiation of computation, to the belief calculated with a complete analysis. This trajectory makes possible the comparative analysis of belief and action, dictated under bounded resources, and ideal belief and action that would be computed if sufficient time for a complete analysis were available. In Section 6.4, we shall review in detail the information presented in each window.

6.1 Architecture of Protos

The Protos system has four major components, pictured schematically in Figure 6.2: (1) an EVC metareasoner, (2) an inference base containing probabilistic inference algorithms, (3) a large knowledge-base of probabilistic relationships, represented as belief networks; and (4) a problem-specific decision problem. The metareasoning component makes use of knowledge about future distributions expected over target probabilities with the use of an algorithm in the inference base to determine the ideal amount of computation time that should be allocated to inference within the

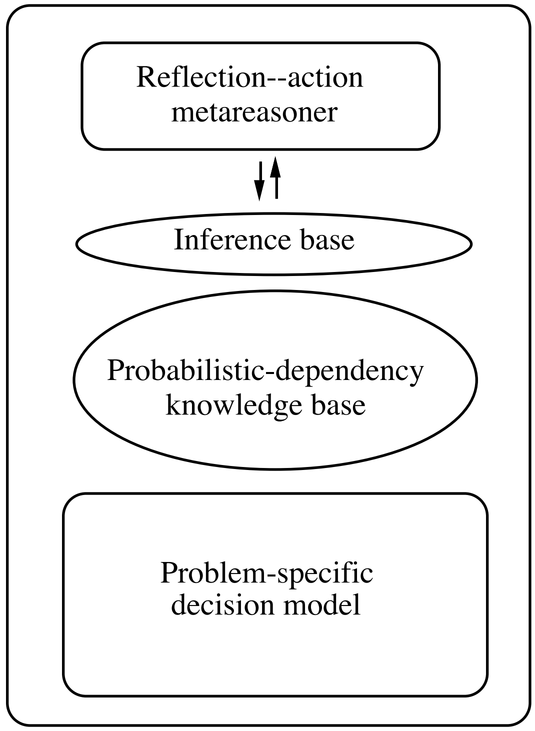


Figure 6.2: The architecture of Protos.
The Protos system has four major components, portrayed schematically from top to bottom: (1) an EVC metareasoner, (2) a base of one or more probabilistic inference algorithms, (3) a probabilistic knowledge base in the form of a belief network, and (4) a patient-specific decision model.

probabilistic knowledge base. Knowledge about the problem instance and about speed with which a problem is expected to be solved is computed and passed to the metareasoner at run time. Protos also adds the time it requires for metareasoning to the total time used in evaluating time-dependent utilities.

As indicated in Figure 6.3, for medical decision making, a patient-specific problem is passed to the Protos reasoner. This problem-specific decision problem contains (1) possible actions and states of the world—these actions and states define possible outcomes; (2) time-independent utility of instantaneous action; and (3) the time dependency of different outcomes that represent the loss (or possible gain) in the utility of an action with time. In the current system, appropriate actions, states of the world, and time-dependent utilities are predefined and passed to the system

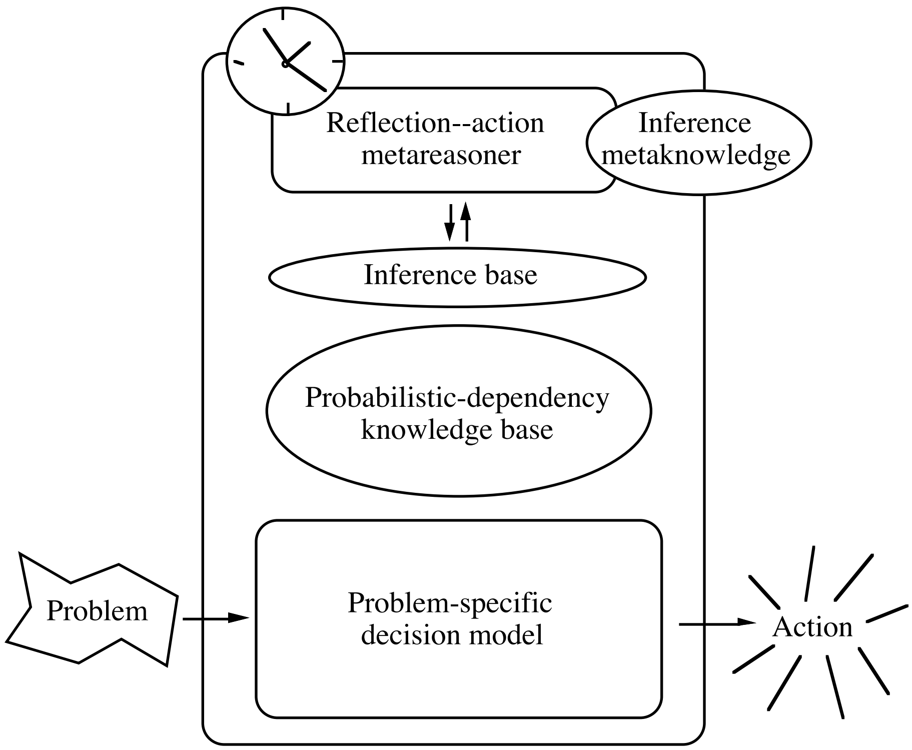


Figure 6.3: Instantiation of Protos with a patient-specific decision problem. A set of outcomes and utilities of those outcomes specific to the case at hand is passed to the Protos reasoner. The problem also includes time-dependent utility information that defines the criticality of taking action. The system applies knowledge about the convergence of a probability distribution over beliefs computed with a belief network to decide on the optimal time to reflect on a problem before recommending or taking action.

as a compact frame of information. I have compiled and stored decision problems that represent prototypical diagnostic dilemmas. However, the patient-specific decisions, outcomes, and time-dependent utilities could be generated by computer-based reasoning methods. I shall describe this feasible extension in Section 6.5.

6.2 Protos Decision Making

Protos computes the probability of propositions required for decision making by performing inference in a large belief network. The sensitive states are defined by the patient-specific decision information that is passed to Protos. Protos computes the

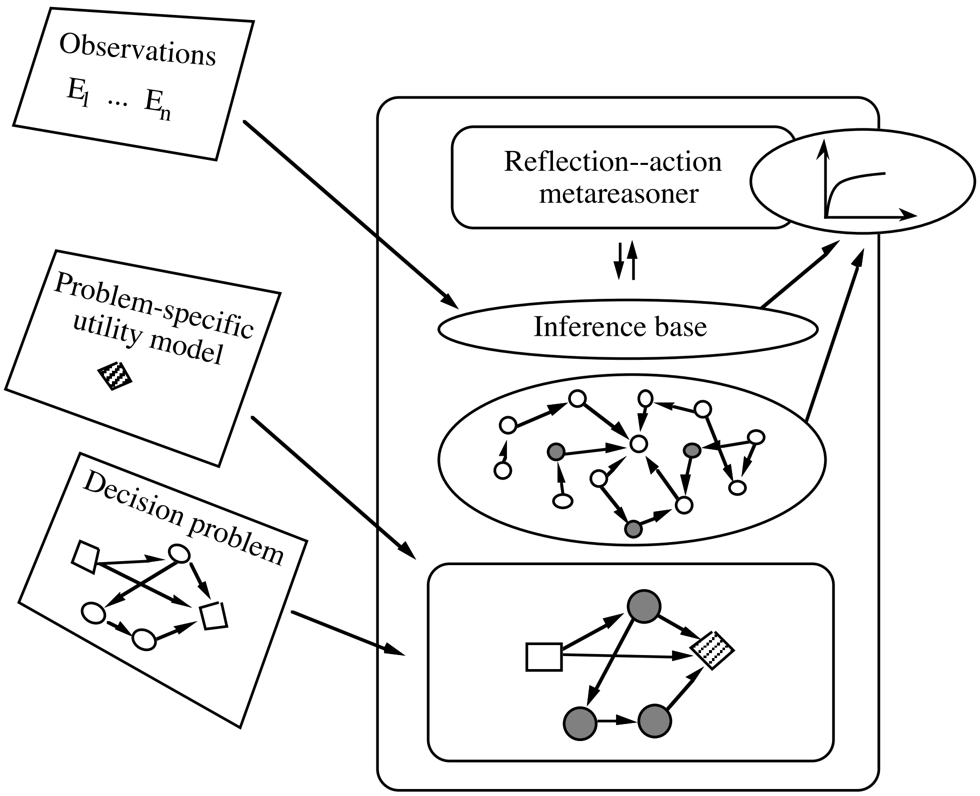


Figure 6.4: Use of metaknowledge about the refinement of $p(\phi)$.
At run time, the decision problem and information about one or more salient observations are analyzed by Protos' EVC metareasoner. The metareasoner makes use of metaknowledge about the progression of probabilistic inference. This metaknowledge is generated dynamically or is accessed from a database of metaknowledge.

EVC of inference as described in Chapter 4, employing an incremental analysis that weighs the costs of dwelling on an inference problem, for a predefined additional quantity of time $\Delta t = T$ before action, with the benefits of additional computation. In operation, the incremental EVC computation involves interlacing probes for positive values of EVC(S, T) and performing additional inference with a belief network.

I shall describe the use in Protos of the bounded-conditioning method (discussed in Chapter 5). Protos uses knowledge about the diminishment of the interval between upper and lower bounds to compute the EVC/BC, described in Section 4.4. Recall that the EVC/BC makes use of information about the expected reduction with computation of the interval between upper and lower bounds on a probability of interest. Such knowledge is made available by bounded conditioning. The current version of Protos reasons about sets of independent binary decision problems. In each of these problems, the system considers two actions, two hypotheses, and the four resulting outcomes. (I shall discuss the extension of Protos to the case of multiple decisions and multiple states of the world in 6.5.) One type of binary decision problem is the treat–no treat decision, where we examine the decision to act now versus continuing to delay. In another form of binary decision problem, we limit the world to the case of For alternative p^* boundary conditions, the solution of EVC/BC for a binary decision problem yields closed-form quadratic functions that report the EVC as a function of (1) the utilities for each of the four outcomes at time t, (2) the current bounds on ϕ $[p(\phi)]$, and (3) the expected convergence of bounds with time.

6.2.1 Iterative EVC Analysis

With the use of bounded conditioning, Protos iteratively applies an EVC/BC analysis, to consider the benefits of additional computation. In particular, Protos examines the value of solving an additional polytree subproblem. Recall from the discussion of bounded conditioning in Chapter 5 that additional computation tightens the bounds on probabilities of interest by an amount equal to the likelihood, or *weight*, of the next subproblem instance.

A view of the skeletal operation of Protos is displayed in Figure 6.5. At each step of the EVC analysis, the latest computed probabilities and knowledge about the likelihood of future probability distributions are considered by the metareasoner to determine whether additional computation has positive net value. If the EVC is positive, the next most relevant subproblem is solved and the distribution over belief is updated. The EVC analysis then focuses on the value and costs of solving the next subproblem. Cycles of *inexpensive* computation of EVC followed by the more *expensive* solution of the next probabilistic subproblem continue until action is indicated by a nonpositive EVC. At this time, the system recommends taking the action that has the greatest utility. This volley of reflection and inference, followed by action in the world, is captured by the cycle in Figure 6.5.

The incremental analysis allows Protos to make use of information about the latest probability distribution calculated. As the value of continuing to compute depends on the constraints imposed by the current belief, using the most accurate result is crucial in making an accurate assessment of computational value.

6.2.2 Extending the Horizon of EVC Analysis

The incremental EVC/BC computation in Protos is myopic or *greedy* in that the contribution and cost of solving only the single next subproblem are considered. The greedy solution has desirable and undesirable features. Its primary advantage is efficiency: the tractable closed-form solution does not impose significant computational burden on the base level. Also, the value of EVC is sensitive to the most recent probability distributions calculated over time; it could be wasteful to expend great quantities of computation on global EVC analyses when new information, useful for updating the EVC estimate, becomes available with computation. Finally, my empirical experience with the use of Protos on medical decision problems has indicated that myopic EVC analyses typically report the same result that would be reported by less myopic analyses. That is, the EVC frequently diminishes monotonically; the EVC tends to remain zero or negative following an initial nonpositive result. In cases where the EVC has a nonmonotonic trajectory, the EVC often becomes nonpositive shortly following the first nonpositive EVC result. Unfortunately, this is not always the case; there are cases where EVC becomes positive after dipping to a nonpositive value. Thus, the myopic, one-step analyses can overlook a positive EVC that lies just

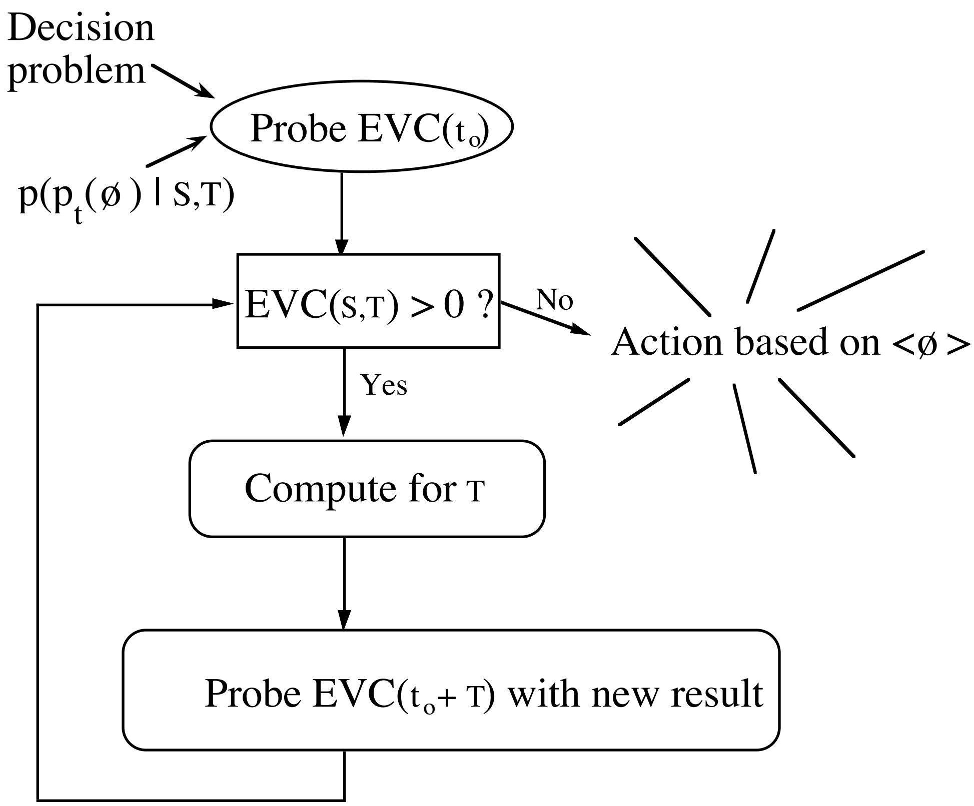


Figure 6.5: Protos iterative EVC analysis.
Protos iteratively evaluates the value of allocating additional time for reflection about a problem. At each step of the EVC analysis, the most recent beliefs and relevant meta-knowledge about the probability of future probability distributions are applied to determine whether that additional expenditure of time has positive net value. The process continues until the system determines that future reflection has a negative expected value. At this point, the system recommends taking the action that has the greatest value.

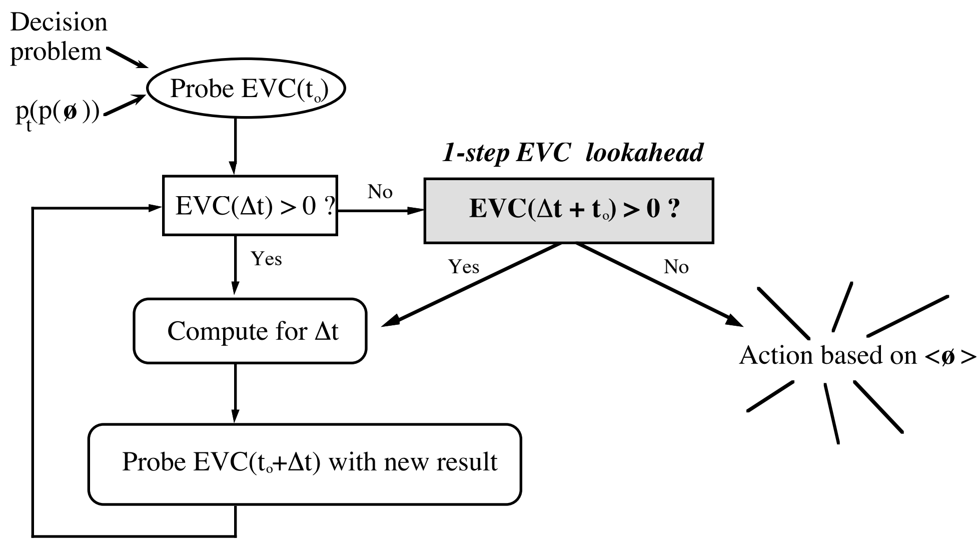


Figure 6.6: Beyond single-step EVC analysis.

Protos has the ability to look beyond the costs and benefits of solving a single subproblem by computing an EVC of solving additional subproblems. In this partial lookahead analysis, the current probability distribution is assumed as a constraint on future distributions.

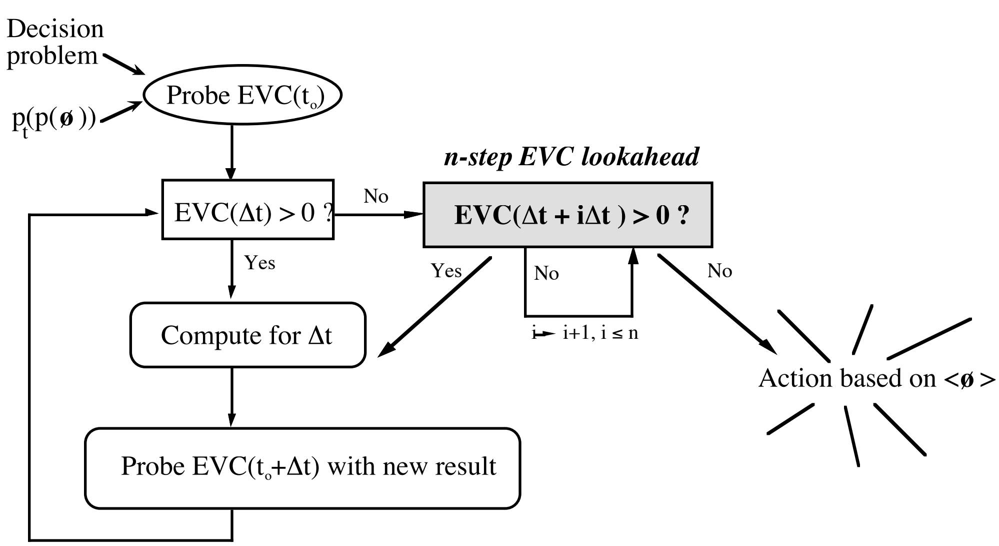


Figure 6.7: More general lookahead for EVC in Protos.

The lookahead analysis has been generalized to an n-step analysis. If the current EVC is nonpositive, the system explores the EVC of solving additional subproblems, in order. If a positive value is found, the system solves a single subproblem and reexamines the future EVC with the constraints imposed by the new bounds on the probability. Otherwise, the system directs action based on the current mean.

beyond a negative swing in EVC.

To identify some of the inaccuracies arising from the myopia of the EVC/BC evaluation, I implemented a partial lookahead EVC analysis in Protos. This partial lookahead is still inexpensive, requiring a quantity of time that is a constant factor greater than the time required for the single-step myopic EVC/BC analysis. In the partial lookahead approach, we evaluate the value of tightening the bounds by solving more than one subproblem. As portrayed in Figure 6.6, if the system determines that the EVC of solving another subproblem is negative, the system repeats the analysis for bounds that will result from solving the *two* next subproblems. This lookahead has been generalized to an n-step lookahead analysis that continues to examine the EVC of up to n additional subproblems, until a positive EVC is found. This approach is displayed in Figure 6.7. If a positive value is found, the system solves one more subproblem, then reexamines the future EVC with the constraints imposed by the new bounds on the probability of interest. If the EVC is still negative after all n of the subproblems have been investigated, the system recommends action in the world based on the best decision available.

6.2.3 Reflex Responses in Protos

The single-step EVC/BC analysis is an inexpensive means of approximating the EVC of probability-bounding algorithms. However, metareasoning analyses may not always be inexpensive. As we design more sophisticated metalevel analyses, we may wish to construct a multilevel approach to metalevel analysis, where a preliminary inexpensive evaluation is performed to screen for a high-likelihood of negative EVC, before we employ a more costly, more precise EVC analysis. A multilevel EVC approach would give a system the ability to react to evidence almost immediately with a *reflex* action, avoiding the cost of a metareasoning analysis.

I constructed a two-level metareasoning facility that allows Protos to forego computation of the EVC if there is information warranting the system to default to a reflex action, based on the current mean of belief or on a compiled recommendation for action, before performing any additional analysis. The reflex capability in Protos demonstrates how expensive components of a metareasoning problem could be avoided in critical situations.

The rationale for a two-level metareasoning analysis for bounded conditioning is based on the cost of preparing a set of updated instance weights to serve as interval-convergence information. This cost is incurred by reasoners using bounded conditioning, in the absence of a set of compiled instance weights. Protos' EVC metareasoner makes use of these updated weights of evidence to reason about future interval between an upper and lower bound on a probability of interest. As I described in Chapter 5, a reasoner must first update the a priori instance weights, by conditioning the old weights on new observations. The computation of new instance weights requires a single multiplication for each instance, and a sum and division for normalization of the probability assignments. Although this process is typically rapid, it is more expensive than are individual incremental EVC/BC analyses that follow the initialization of the weights. I developed a two-stage metareasoner that allows Protos to react before committing additional time to the updating of its weights, based on a preliminary EVC *screening analysis*. This preliminary analysis reduces the *minimum response time* required by the system before acting, and demonstrates how evidence about a context could be used to usurp costly metalevel deliberation.

Two inexpensive EVC-screening approaches were implemented: (1) a worst-case EVC analysis, based on the worst-case convergence properties of bounded conditioning for a belief network (as described in Chapter 5); and (2) an average-case EVC estimate, based on empirically derived sequences of instance weights. In both cases, Protos is told which probabilistic knowledge base is being used, and applies the worst-case or average-case convergence EVC analyses. If the preliminary analysis is positive, Protos expends the overhead in updating the instance weights, and performing a complete meta-analysis. In the use of the average-case EVC, the actual weights are recorded and averaged with the convergence seen so far, to be used in future average-case analyses. The preliminary EVC screening approach is captured by the schematic in Figure 6.8. The current version of Protos uses the average-case analysis to choose between taking a reflex action and doing a more involved meta-analysis. As we shall see in Section 6.4, the EVC-screening approach in Protos reduces the minimum time required by the system for a response, and demonstrates how more

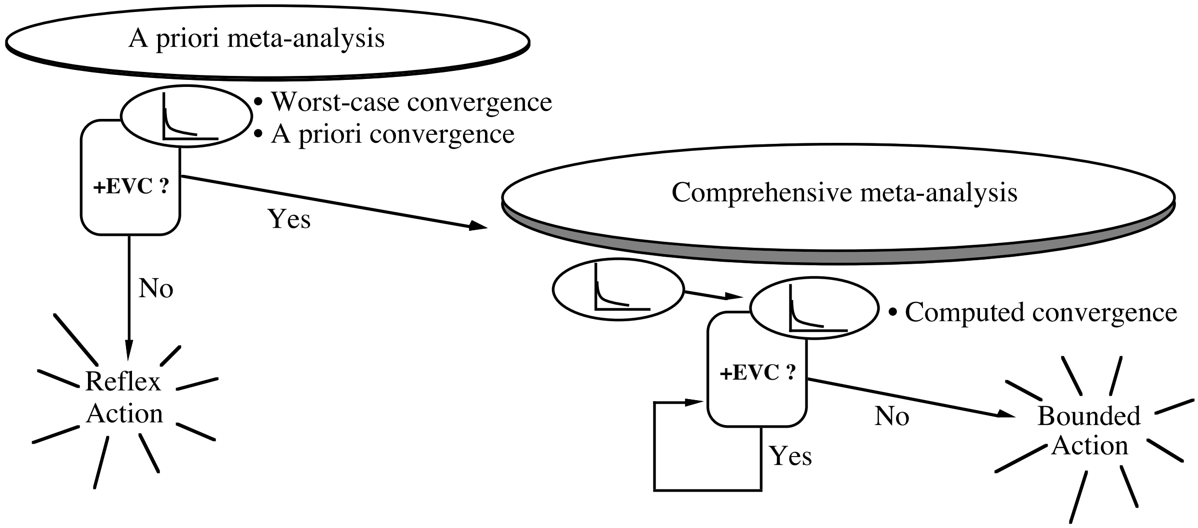


Figure 6.8: Reflex action in Protos.
Protos applies a preliminary average-case EVC analysis at the outset of a case to determine whether to react immediately with a reflex action or to allocate further resources to prepare the instance weights for a more accurate EVC computation. The preliminary screening approach reduces the minimum time required by the system for a response, and demonstrates how more complex metareasoners might be partitioned into several tiers of analysis.

complex metareasoners might be partitioned into several tiers of analysis, each tier of meta-analysis requiring more time and yielding more accurate EVC estimates.

6.3 Time-Dependent Utility in Protos

As highlighted in Chapter 4, the cost associated with reasoning is central to the calculation of EVC. We shall now examine the representation of time-dependent utility of alternate outcomes in Protos, and shall discuss work on the dynamic formulation and customization of time-dependent utility models.

6.3.1 Representation of Time-Dependent Utility

In Protos, costs of delay are represented as linear and exponential decay functions of the value of alternative outcomes. A cost function can be specified for any relevant outcome. The nature of the function and constant parameters are stored in a frame structured by outcome. The structure of the file used to represent time-dependent utility information is displayed in Table 6.1. Individual outcomes considered in a current decision problem are listed in the first column. In this case, the two actions, *Action 1* and *Action 2*, and two states of the world, *Hypothesis 1* and *Hypothesis 2*, lead to four possible outcomes, listed in the first column. For each outcome, the frame contains (1) the utility of that outcome given immediate action (at time t_α), (2) a functional form and parameter describing the change in utility of each outcome with a delay, and (3) the numeric or symbolic description of the value that the utility of an outcome will converge to with time. For assessment, the initial time t_init is defined as the moment a state of the world becomes true, or as an estimation of this time, based on when salient manifestations have been observed.

Protos allows several functional forms for utility decay and deadline. These include the linear and exponential forms,

$$u(A_i, H_j, t) = u(A_i, H_j, t_\alpha)e^{-k_a t}$$

$$u(A_i, H_j, t) = u(A_i, H_j, t_\alpha) - c_b t \quad \text{where} \;\; u(A_i, H_j, t) \geq 0$$

where k_a and c_b are assessed parameter constants. In the example, the time dependency of the utility of taking action Action 1, given state Hypothesis 1, is described as decaying from its initial value of 0.25 at t_α by a negative exponential function, with a decay constant of 0.006.

In many cases, the time-dependent utilities of outcomes converge on a value of zero with time. However, at other times, a decision maker may wish to specify a lower bound on the utility. Protos allows a user to specify a lower bound on a time-dependent utility as a numeric quantity. In addition, the utility of an outcome can be defined to converge to the value of another outcome. For the time-dependent utility model displayed in Table 6.1, the utility of taking action Action 1, given that Hypothesis 1 is true, converges to the utility of taking action Action 2, given Hypothesis 1.

Outcomes		U_{t_α}	Time Dependence		Convergence
Action 2	Hypothesis 2	1.00	Ø	–	1.00
Action 1	Hypothesis 2	0.65	Ø	–	0.65
Action 1	Hypothesis 1	0.25	Exp	0.006	$u(A_i, H_j)$
Action 2	Hypothesis 1	0.02	Ø	–	0.02
	○	○	○	○	○
	○	○	○	○	○
	○	○	○	○	○

Table 6.1: Time-dependent utility information is represented in Protos as a structured file of information containing (columns from left to right): (1) a list of actions and outcomes, (2) the utility at t_α of alternate outcomes, (3) the functional form and parameter of the cost of delay; and (4) the value to which the utility of each outcome will converge. The latter value can be stated in terms of the utility of another outcome.

6.3.2 Assessment and Construction of Utility Models

Time-dependent utility models for alternate decision problems can be directly assessed by the procedures described in Section 4.2.3 and cached for use in real time by Protos. The use of prototypical utility models and cost functions in machine reasoners suffers from the same problem that plagues professionals charged with making decisions for their clients. In practice, professionals—such as physicians—do not acquire detailed knowledge about the preferences of their clients. Instead, they make a best guess about the preferences of the people they are serving. Several researchers have investigated the real-time assessment of a decision maker's preferences (McNeil et al., 1982; Barry et al., 1988; Jimison, 1990). In one approach, medical decision analysts encoded several prototypical utility models, and used attributes of the patient's personality to choose the most appropriate model for that patient.

To customize the utility models used in Protos to specific situations and to different individuals, I investigated the automated construction and customization of time-dependent utility models. We can view utility model construction and refinement as a process of diagnosis from evidence. In the general case, we may not be certain of a patient's preferences; thus, determining the utility of alternate outcomes is a problem of reasoning under uncertainty. I chose to perform utility construction as a case of deterministic diagnosis. A Protos subprogram named Puma (for Protos Utility Modeling and Assessment) was designed to construct and customize time-dependent utility models as a function of patient vital signs including a patient's age, heart rate, blood pressure, and partial pressure of oxygen in the blood. Puma allows an expert clinician, or other principal agent for a patient, to specify object-level and time-dependent components of a preference model with functions that take as arguments key attributes that characterize a patient's personality or physiology. In Protos experiments, an expert emergency-room physician, who played the role of a principal agent for a patient in a critical setting, specified functions which dictate the cost of delay as a function of a patient's vital signs.

6.4 Protos in Action

As indicated in Figure 6.9, the system decides on one of three basic reasoning strategies. Depending on the probability distribution over the outcomes, the time-dependent utilities, and the metaknowledge about the expected convergence of probabilistic inference, Protos may (1) recommend immediate action after a precursory analysis, (2) dictate a best action under bounded resources after some partial inference, or (3) demonstrate the dominance of one of the possible actions after performing inference long enough to prove the optimality of that action. A recommendation for immediate action is based on a preliminary average-case EVC analysis. The recommendation of a best action under bounded resources is the decision with the greatest utility at the time the EVC becomes zero. Finally, if a computed upper bound on a probability drops below a decision threshold, or a lower bound rises above that threshold, before the EVC becomes nonpositive, then a best decision is proved. As we shall see, these decisions, and the corresponding actions in the world, typically are made well before inference would be completed.

Let us now examine Protos' deliberation about inference within the ALARM belief network, which was introduced in Chapter 4 (Figure 4.4). I shall describe the case of dominant decisions. In Chapter 7, we shall examine all classes of Protos decision

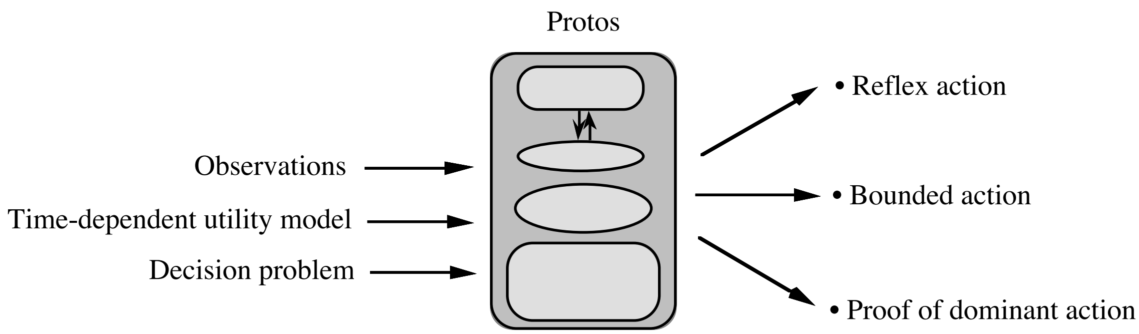


Figure 6.9: Protos actions under varying resource constraints.
Depending on the time-dependent utilities and stakes of the decision problem and the metaknowledge about the expected convergence of probabilistic inference, Protos determines one of three possible actions: (1) recommend immediate action after a precursory analysis, (2) prove the dominance of one of the possible actions, or (3) dictate the best action under bounded resources.

making, as part of the validation of the Protos system on another belief network.

Suppose the utility model, represented by the first four rows of the frame in Table 6.1, describes the relevant actions, stakes, and time-dependent utility in the ICU decision context described in Chapter 4. Let us see how Protos integrates the utility information into decisions about computation. Figure 6.10 shows Protos' display of the convergence of the upper and lower bounds (ub,lb) on a probability of state H_1 (e.g., respiratory failure) with time, with the mean of the assumed uniform distribution determined by these bounds. The figure also shows the time-dependent decision threshold, $p^*(t)$. The level and time-dependent changes of p^* are a function of the utilities and decay of the outcomes specified in the utility frame. Vertical hash marks on the x axis represent the solution of individual problem instances. There are 108 instances generated by this cutset.

Figure 6.11 shows Protos' decision about an ideal time for halting inference. Notice that it is ideal to halt and to act to treat the patient after analyzing only six of the 108 subproblems. As the system notes at the bottom of the graph, each bounded-conditioning subproblem requires about 10 seconds.[1] After dwelling on the problem

[1]The solution time depends on the implementation and hardware. Recent engineering enhancements approximately doubled the speed of subproblem analysis for the ICU network.

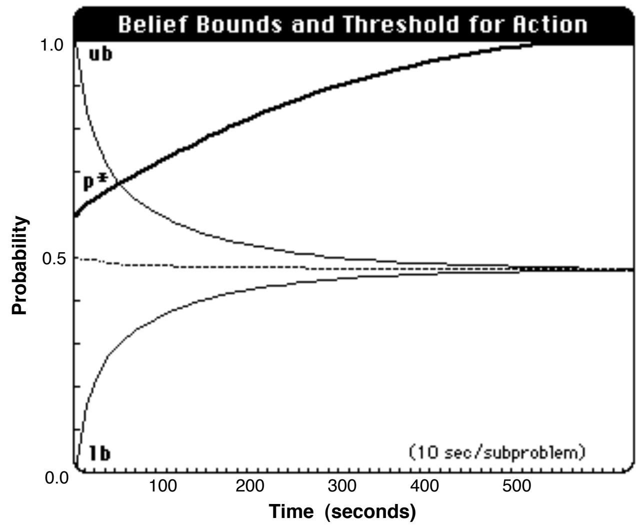


Figure 6.10: A belief update within the ICU belief network.
Protos displays the convergence of the upper and lower bounds (ub, lb) on a probability of interest with time, and the mean of the distribution, assuming uniformity. It also displays the time-dependent decision threshold (p^*). Vertical marks on the x axis represent the solution of individual problem instances. There are 108 instances generated by this cutset.

for 672 seconds, Protos reports in a short message that it has proved that the best decision is A_1 (e.g., to act to treat respiratory failure). This report refers to the observation that the upper bound of the probability falls below the decision threshold *before the EVC/BC becomes nonpositive.*

Let us examine the EVC, computed iteratively before the solution of each subproblem. Figure 6.12 shows the value of the EVC/BC as inference progresses. The graph of EVC/BC is scaled for each case so we can inspect the structure of small changes in EVC/BC. Given the information in the utility frame, the current positions of the upper and lower bounds, and the expected convergence of the bounds with time, the EVC/BC rises to a maximum before decreasing to zero as the upper bound of the probability passes beneath the value of p^*. Figure 6.13 superimposes the object-level EVC/BC on the same graph. This quantity is the value of reasoning in a world where computation is free. We consider all the utilities to be time independent for calculating the object-level EVC/BC.

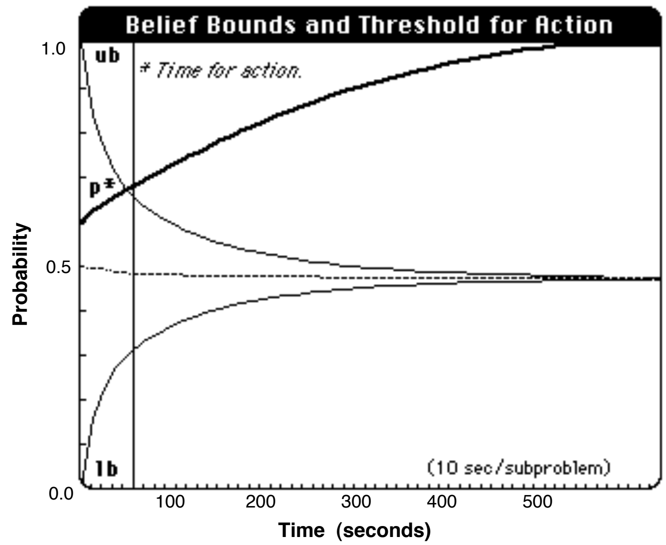


Figure 6.11: An ideal halting time.

Protos notes that the upper bound of the probability of interest dips below the decision threshold after it solves six of 108 subproblems.

Protos also displays the metaknowledge it is using to compute the EVC. Figure 6.14 is a histogram that shows the contribution that the solution of each subproblem will have to the convergence of the bounds. Recalling details about bounded conditioning from Section 5.2, we determine such a contribution by computing the *weight* of each subproblem. Weights are computed by updating the prior weights on instances with evidence and sorting these subproblems by their contribution. The bounds converge with the solution of each subproblem at a rate indicated by the height of the histogram at that subproblem.

Protos also exhibits the value of the utility of the outcomes under consideration. This information for the current case is displayed in Figure 6.15. Notice that, as dictated by the utility frame in Table 6.1, only outcome A_1, H_1 (the utility of acting to treat the patient for respiratory failure) is described by a time-dependent utility. We can easily make other utilities time dependent by adding a decay function and parameter to the utility frame.

Finally, Protos shows us a graph, displayed in Figure 6.16, of the initial utilities of the outcomes at t_α, the utilities of the outcomes at the ideal halting point, and the

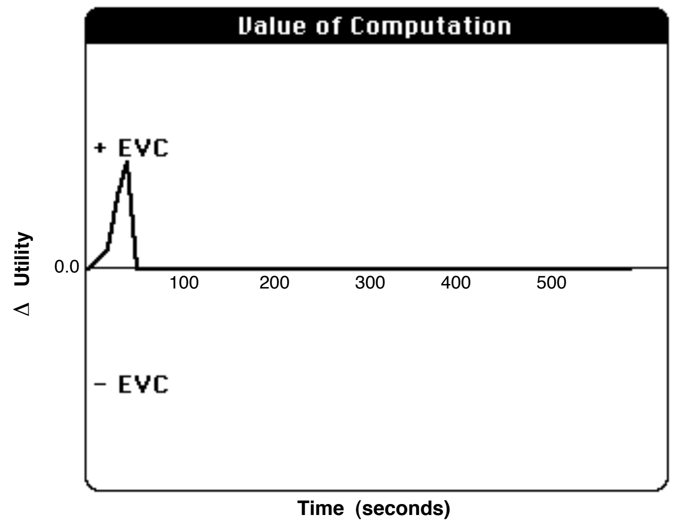


Figure 6.12: The EVC/BC of inference with bounded conditioning.
This graph shows the EVC/BC associated with the solution of the "next subproblem." Note that, in this case, the EVC rises before beginning a steep descent to zero as decision dominance is proved.

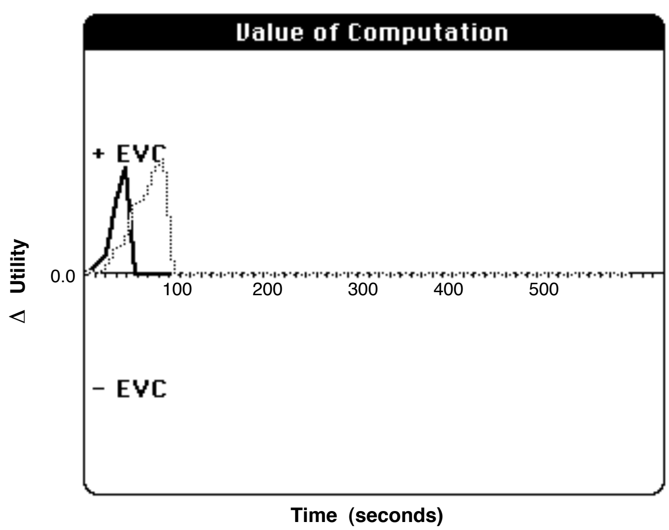


Figure 6.13: Comparison of the comprehensive EVC/BC with object-level value.
The lighter graph represents the object-level value of inference, the value of computation in a world where delay incurs no cost.

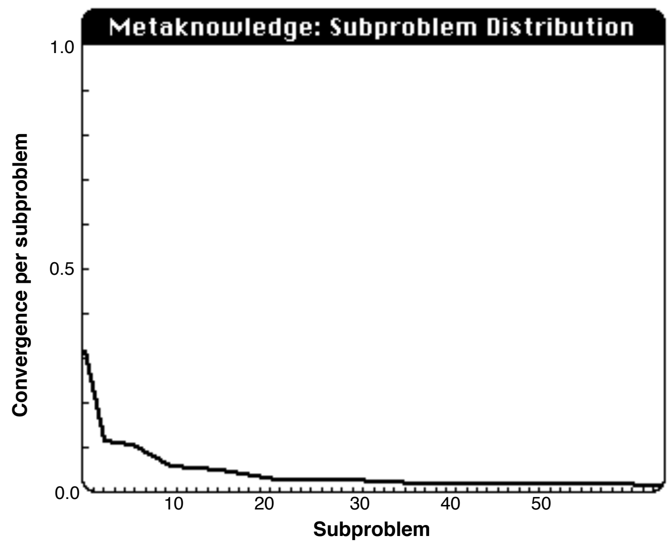


Figure 6.14: Knowledge about the convergence of an interval.
This graph displays a histogram that characterizes the contribution of each subproblem to the convergence of the bounds interval. Protos computes the graph by updating the prior weights on instances with evidence and sorting the subproblems. The bounds converge per subproblem analyzed at a rate dictated by the height of this graph over each subproblem (representing that problem's weight).

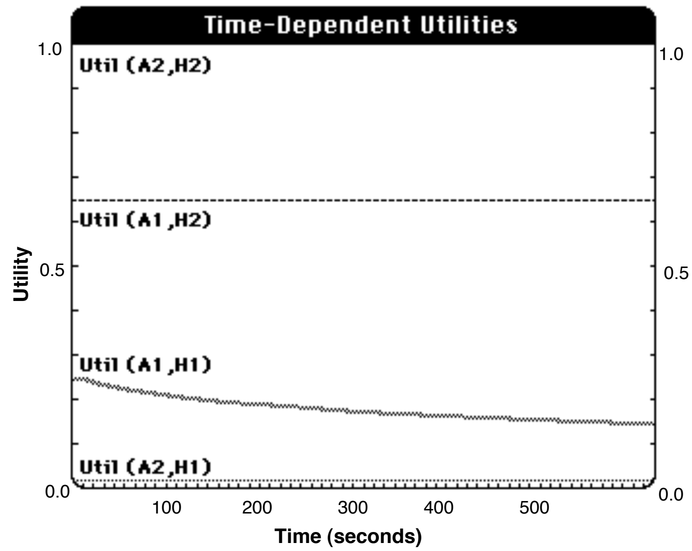


Figure 6.15: The time-dependent utilities of four outcomes.
In this case, only outcome A_1, H_1—taking action to treat a time-dependent state—is decaying with time. This change in utility is responsible for the time-dependent decision threshold (p^*).

expected utility of the alternative actions being considered (A_1 and A_2) as a function of the probability of a hypothesis ($p(H_1)$). The solid lines are the utilities of action at the time of halting. Broken lines indicate the initial utilities of the actions as functions of the probability of a hypothesis.

The system can be instructed to superimpose the current probability distribution on the graph of the utilities of actions. As pictured in Figure 6.17, with the use of a bounding algorithm like bounded conditioning, we post the lower and upper bounds on the probability that will be computed with sufficient time for a complete analysis. We see also the relationship of the decision threshold p^* to the current belief. The approximate probabilistic calculation, in conjunction with information about the utility of alternative actions, may be viewed as a partial result for decision-theoretic inference. Protos can complete the analysis and post the final probability computed with sufficient computation. This probability is displayed in Figure 6.18.

Protos can demonstrate the sensitivity of the optimal time to compute before acting in the world. Such changes in optimal halting time and action, depending

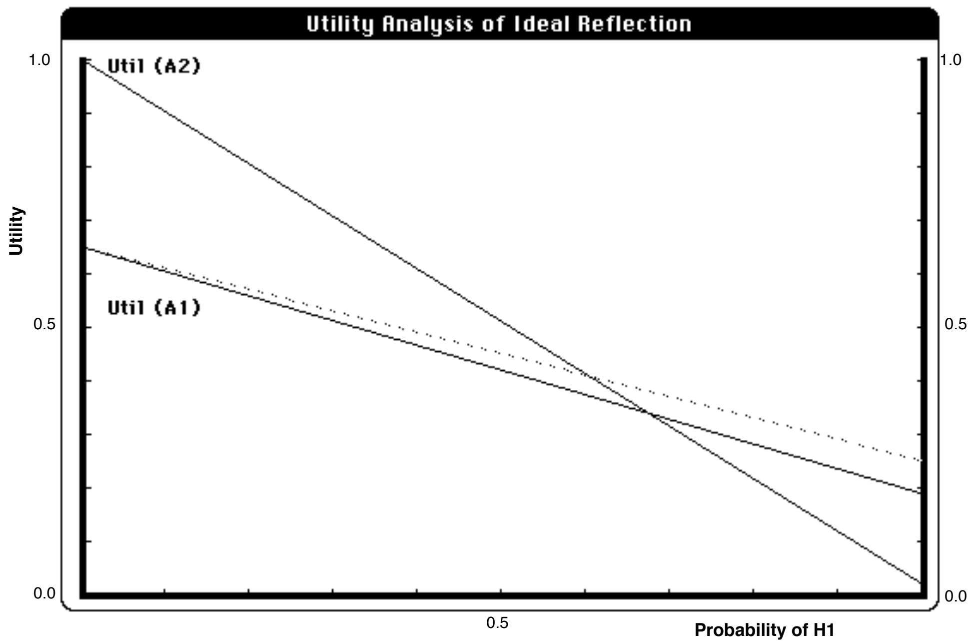


Figure 6.16: Time-dependent value of outcomes as a function of probability. Protos displays the initial utilities of the outcomes and the utilities at the ideal halting point. The system graphs the expected utilities of two actions (A_1 and A_2) as a function of the probability of a state ($p(H_1)$) and shows how the expected utility of taking action (A_1) diminishes with delay. The solid lines are the utilities of action at the time of halting. The broken lines adjacent to the solid utility lines indicate the utility of acting at t_α.

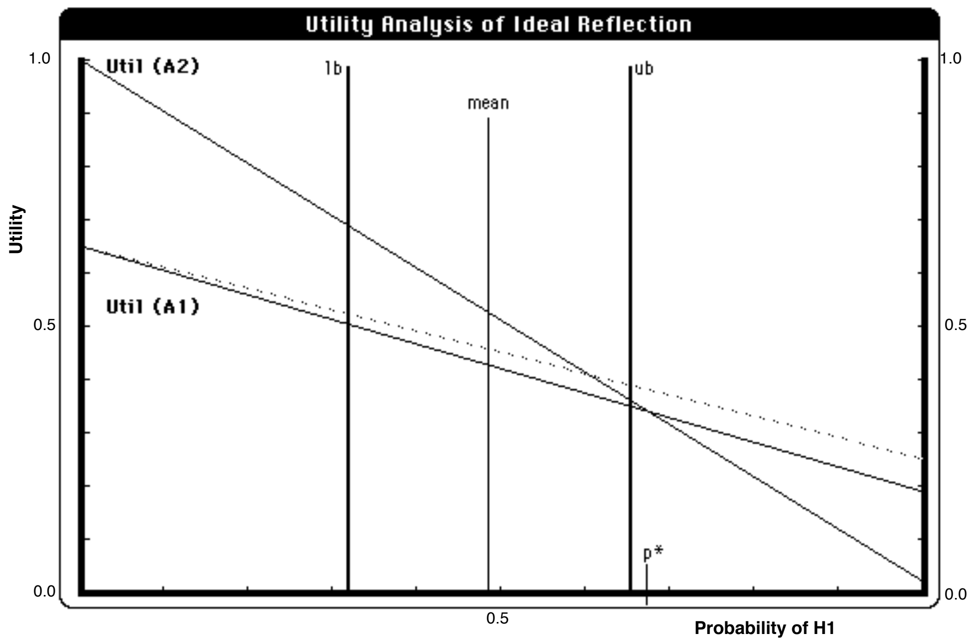


Figure 6.17: A partial state of information.

Protos displays the probability distribution at the time of halting. Upper and lower bounds on a final probability, and the mean of an assumed uniform distribution are displayed. The decision threshold, p^*, also is displayed. We see that the upper bound is below the revised p^*. The expected utility of the action is the height of the dominant action at the mean of the current distribution.

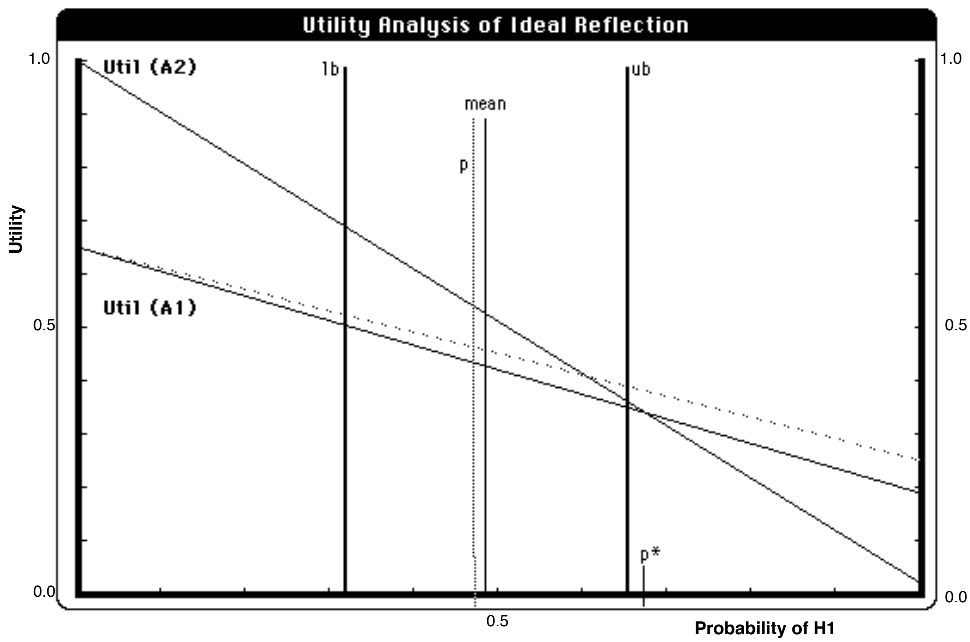


Figure 6.18: Location of the final result.

Protos also can show us where the final probability (ϕ) is located (dotted line) after exhaustive computation. In this case, the final probability is very close to the current mean of the bounded-conditioning approximation.

on the parameter used to describe the time-dependent utility of outcome A_1, H_1 are demonstrated in Figure 6.19. Here, we increase the rate at which the utility of action A_1, given hypothesis H_1 is true, decays with time. We see that the ideal reasoning time is decreased. Figure 6.20 shows how *decreasing* the rate of decay of the utility of A_1, H_1 increases the time we should continue to compute before acting.

Let us now examine another update of $p(H_1)$, given new observations. Figure 6.21 (a) shows the upper and lower bounds of the new update. In addition, the graph displays the time-dependent p^* and ideal halting time, based on the utility model assumed for the previous example (Figure 6.20). The system recommends action A_2 after computing for 112 seconds. Figure 6.21 (b) displays the comprehensive and object-level EVC/BC.

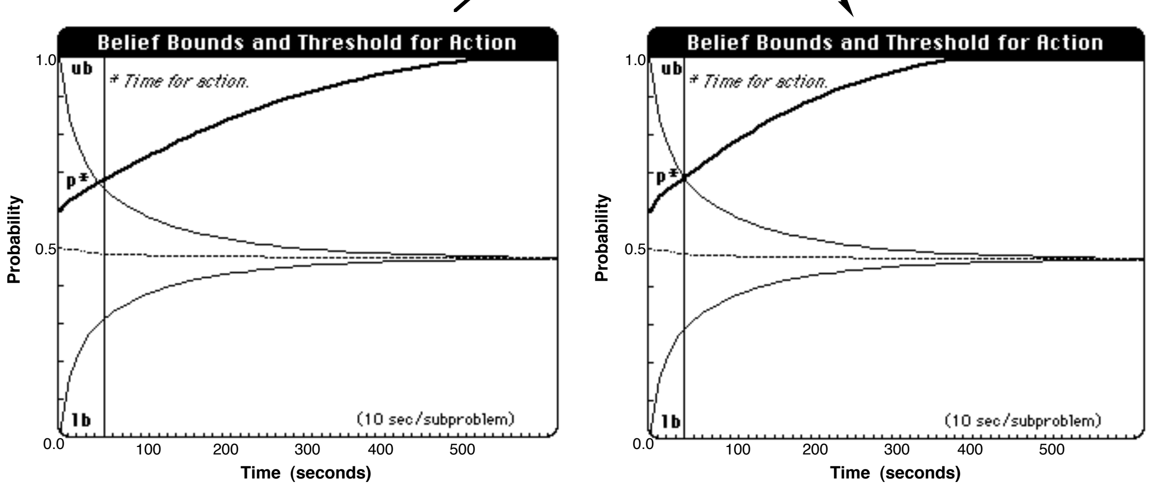
Now, we shall revise the utility model to see how the halting time and action are affected. We shall analyze the same update with an updated utility model described by the frame at the top of Figure 6.22. Notice that the initial utilities of outcomes A_1, H_1 and A_2, H_1, and the decay of the utility of A_1, H_1, are reduced in the revised model (indicated by the underscoring). With the new utility model, Protos reasons for 232 seconds and makes a decision to take action A_1. *The ideal time to compute and the ideal action to take after ceasing to reflect about a problem depend greatly on the details of the utility model.* Figure 6.23 displays the states of belief and utility at the ideal halting times for the utility models described in Figures 6.21 and 6.22.

In the examples presented in this chapter, we examined cases where Protos proved dominance of a decision by showing that a bound on a probability crosses above or below a decision threshold before the EVC becomes nonpositive. In Chapter 7, we shall see examples of action recommended before a decision threshold is reached.

6.5 Extensions of Protos

Several extensions of Protos are feasible. I shall mention salient areas of research interest. We have focused, in this chapter, on the use of single flexible reasoning strategy. We can apply the EVC formulae, introduced in Chapter 4, to examine the best of several algorithms to apply to inference by exploring the EVC for a set of

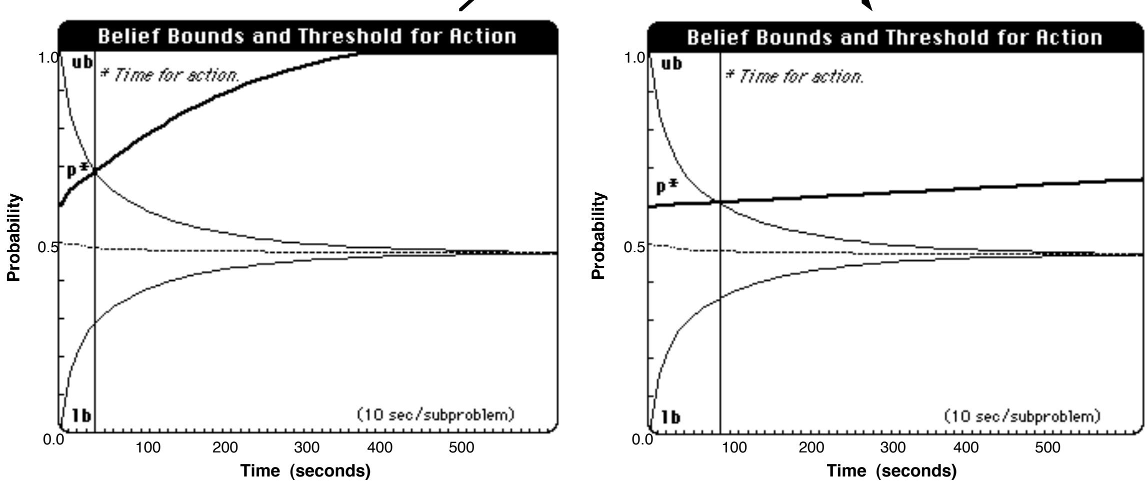
Outcomes	U_{t_α}	Time dependence	Convergence
• Action 2, Hypothesis 2	1.00	Static	1.00
• Action 1, Hypothesis 2	0.65	Static	0.65
• Action 1, Hypothesis 1	0.25	0.006 → **0.008**	$u(A_2H_1)$
• Action 2, Hypothesis 1	0.02	Static	0.02



(a) (b)

Figure 6.19: Increasing the criticality of the decision context.
We represent a context of increased time criticality by increasing the rate of decay of the utility of A_1, H_1. (a) shows the earlier time for ideal action and (b) shows the new time for action indicated by the revised utility model. The ideal time for reasoning has decreased.

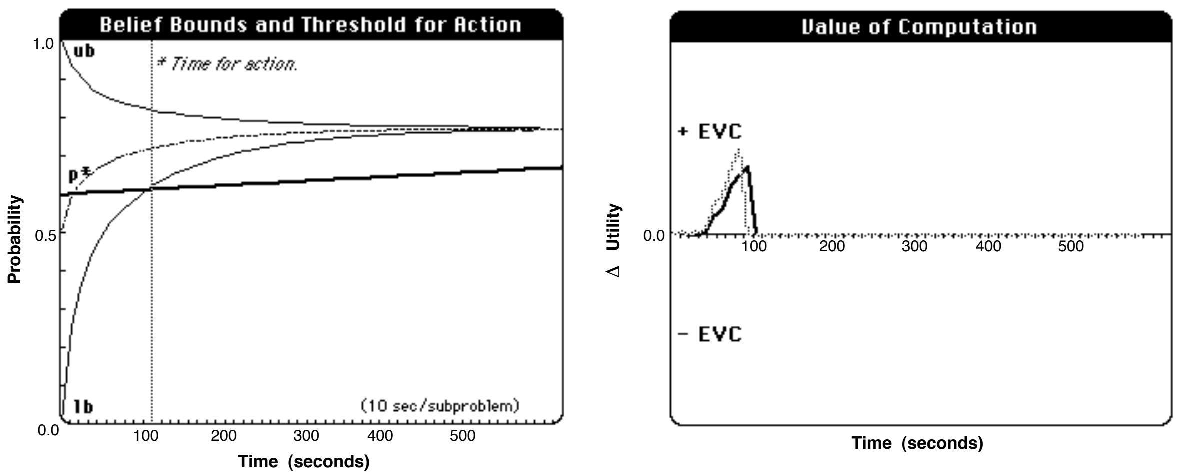
Outcomes	U_{t_α}	Time dependence	Convergence
• Action 2, Hypothesis 2	1.00	Static	1.00
• Action 1, Hypothesis 2	0.65	Static	0.65
• Action 1, Hypothesis 1	0.25	0.008 → **0.0005**	$u(A_2 H_1)$
• Action 2, Hypothesis 1	0.02	Static	0.02



(a) (b)

Figure 6.20: Result of decreasing the cost of delay.

In this case, if we decrease the cost of delay, we have a decision threshold that rises more slowly with time.



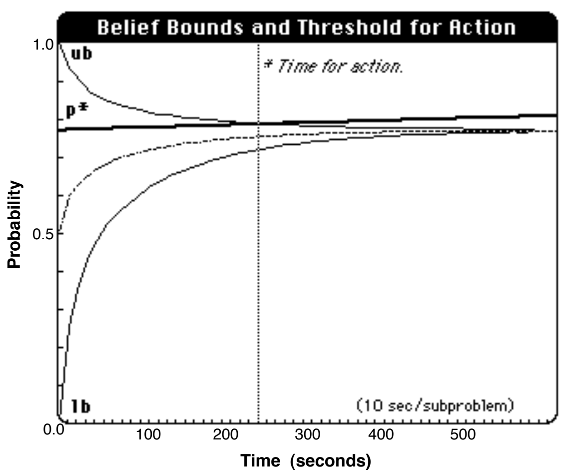
(a) (b)

Figure 6.21: Another probability update.

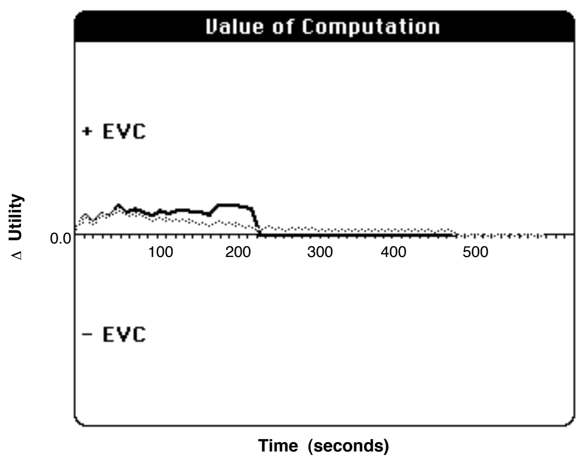
The graph in (a) shows convergence of bounds on the same probability with different evidence, assuming the same utility model that was used in the previous analysis (displayed in Figure 6.20). The figure highlights how an ideal halting time can be sensitive to the location, as well as the rate of convergence of the bounds. The EVC is graphed in (b); the comprehensive EVC/BC (darker) and the object-level EVC/BC (lighter) for this update are displayed.

Outcomes	U_{t_α}	Time dependence	Convergence
• Action 2, Hypothesis 2	1.00	Static	1.00
• Action 1, Hypothesis 2	0.65	Static	0.65
• Action 1, Hypothesis 1	0.25	0.0005	
• Action 2, Hypothesis 1	0.02	Static	

Outcomes	U_{t_α}	Time dependence	Convergence
• Action 2, Hypothesis 2	1.00	Static	1.00
• Action 1, Hypothesis 2	0.65	Static	0.65
• Action 1, Hypothesis 1	**0.20**	**0.0002**	u(A_2H_1)
• Action 2, Hypothesis 1	**0.10**	Static	0.10



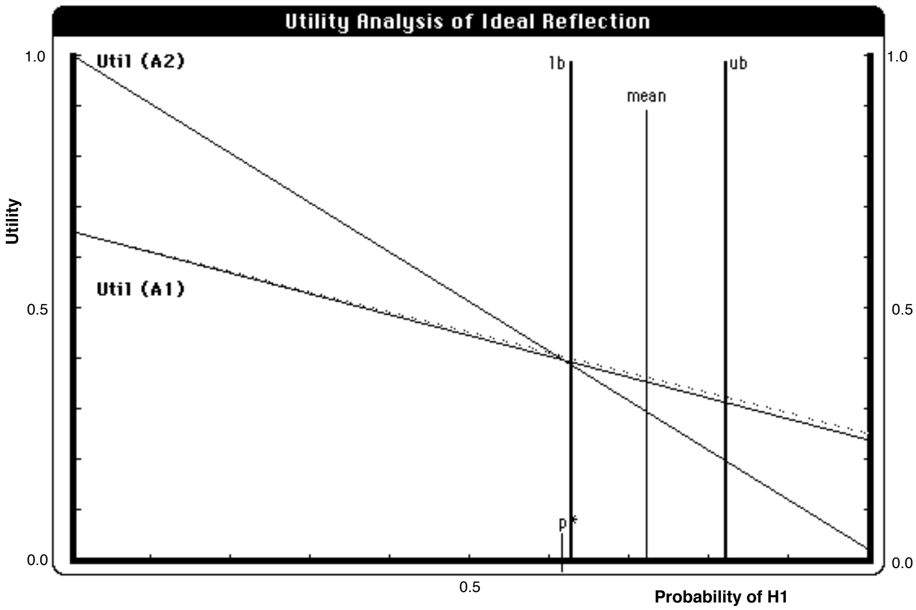
(a)



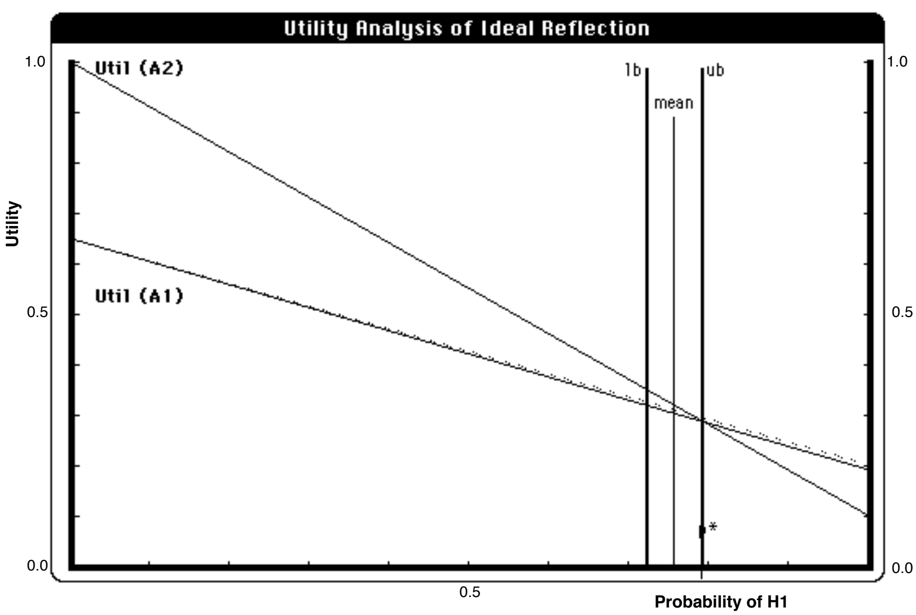
(b)

Figure 6.22: A revised utility model.

As indicated by the underscored numbers in the utility frames at the top of the figure, we update the utilities of outcomes A_1, H_1 and A_2, H_1, and further reduce the time-dependent decrease in the utility of A_1, H_1. The revised time-dependent probability threshold, $p^*(t)$, and the later ideal time for halting, are shown in (a). In Figure (b), the updated EVC/BC (darker line) and object-level EVC/BC (lighter line) are shown.



(a)



(b)

Figure 6.23: A closeup of the information states at the time of action. In (a) and (b), the two different endpoints of inference for the two different utility models described in Figures 6.21 and 6.22 are shown.

representative computation times. I am integrating stochastic-simulation strategies in Protos. We can also extend Protos' analyses beyond binary actions and states of the world. Adding actions is straightforward, and requires few changes in the EVC evaluation. Increasing the number of states of the world leads to a growth of complexity of the EVC analysis. We have been seeking ways to approximate more sophisticated EVC analyses, to make feasible the analysis of action given a consideration of larger number of hypotheses. Another realm of feasible refinement for Protos is the development of more sophisticated lookahead to allow the system to avoid halting inference when great inferential value lies just around the corner, a few subproblems away. Finally, in the current version of Protos, problem-specific utility and decision information is passed to the reasoner. There is great opportunity to marry the techniques demonstrated in the Protos system with a dynamic planning system that can dynamically build models and identify key decisions effectively, depending on a set of observations. Of particular promise is the combination of Protos with planners that identify relevant decisions by identifying tradeoffs (Wellman, 1988).

6.6 Descendants of Protos

Applications of descendants of the Protos system include monitoring systems, autonomous closed-loop systems, and robot decision-making systems. The techniques elucidated in Protos can allow designers of autonomous control systems and mobile robots to employ large, expressive probabilistic knowledge bases for decision making in contexts that might have greatly varying levels of criticality. Such mechanisms could serve as the basis for the use of large decision models in closed-loop medical systems. As an example, the pioneering work by Sheppard and colleagues on the use of closed-loop monitoring and decision making about the infusion of vasoactive agents in cardiac patients (Sheppard and Sayer, 1977; Sheppard, 1980) might be extended with more sophisticated computer-based decision making. An autonomous system employing large, expressive belief networks and normative metareasoning would have the ability to dynamically custom-tailor its response to a wide range of challenges.

Such systems might be entrusted one day with the responsibility for delivering complex therapy regimens to patients.

Beyond the dynamic needs of real-time control systems, the principles of bounded rationality can also apply to the control of "open-loop" decision-support systems. As large knowledge bases and diverse inference-approximation algorithms are developed and enhanced, it may become important for a decision-support systems to make decisions about the length of time to reason before making a recommendation for action. In this context, the principles described in this dissertation may allow a system to recommend preliminary action within several minutes, and to review earlier recommendations as inference progresses.

Chapter 7

Validation of Protos' Behavior

What is the value of using the normative-metareasoning techniques developed in this dissertation? In this chapter, we shall examine the performance of Protos' metareasoning techniques on several decision problems and belief networks. I shall define and illustrate the *value of metareasoning* with the bounded-conditioning algorithm. We shall see that the value of normative metareasoning, for determining ideal reflection and action under time pressure, depends on (1) the time-dependent utilities associated with alternative outcomes, (2) the difficulty of inference, and (3) the time needed for metareasoning. The performance of Protos suggests that an efficient metareasoner can be a valuable asset to a reasoner because of the metareasoner's ability to optimize expensive object-level problem solving, in return for small investments in metalevel computation.

7.1 Value of Metareasoning and Control

We can view all computer-based reasoners as being directed by a metareasoning policy. Programs that do not have an explicit metareasoner assume a default metareasoning strategy. For example, a computer program that solves all problems completely, regardless of the costs of computation, is directed by a default policy of "compute until a solution is reached." We can imagine more sophisticated metareasoning policies. For instance, we might write a computer program that uses a simple heuristic rule to check the size of a problem before solving that problem; if the problem is bigger than some predefined value, the system is directed to take immediate action in the world, foregoing further analysis. However, if the problem is smaller than the predefined value, the computer is directed to solve the problem completely before taking action.

7.1.1 Expected Value of Metareasoning

In contrast to the use of a simple reasoning policy, such as a set of predefined control rules, Protos' reasoning is shaped dynamically by a set of normative metareasoning techniques, based on the computation of EVC. The study of the value of metareasoning techniques requires a comparative analysis of two metareasoning policies. Thus, to gauge the value of Protos metareasoning, we must compare the system's behavior with that of a less sophisticated default metareasoning policy.

I define the *value of metareasoning*, EVM$(\mathcal{M}^i, \mathcal{M}^j, I, \xi)$, as the difference in the utility of employing two different metareasoning strategies, $\mathcal{M}^i$ and $\mathcal{M}^j$, for solving a specific problem instance I. The EVM is sensitive to the performance of the object-level reasoning procedures. In evaluating the EVM, we assume that the object-level machinery is held constant; for probabilistic inference, we fix the number of available inference strategies and the speed of processing. We use ξ in this setting to refer to background information about the object-level procedures and performance of a computer-based reasoner.

I use EVMP$(\mathcal{M}^i, \mathcal{M}^j, \mathcal{P}^k; t, \xi)$ to refer to the more comprehensive measure of the gain in value associated with the application of a new metareasoning policy for solving a *population of problems* $\mathcal{P}^k$ over a period of time t. The EVMP is useful for comparing the relative effectiveness of two metareasoning policies in different contexts. Unfortunately, the EVMP is more difficult to compute than is the EVM because it requires both a probabilistic characterization of the distribution of decision problems that may challenge a reasoner over time and a means of combining a complex history of outcomes. As we shall see in Section 7.2, it is straightforward to determine the

EVM for single decision problems.

7.1.2 Normative Metareasoning Versus Simpler Policies

Let us consider metareasoning strategies that are simpler than the Protos EVC approach. A feasible metareasoning policy, for computer-based reasoners that manipulate probabilities and utilities, is to cease reflection and to take action whenever a bound on a probability passes over a probability threshold. Such a *decision-threshold* policy would direct a reasoner to take action whenever a lower bound on a probability of interest rose above a decision threshold, or an upper bound on that probability became smaller than that threshold. A threshold policy would not require the computation of EVC. The policy would require the facilities of Protos that represent time-dependent utility, that compute changes in the dynamic decision threshold, and that monitor the relationships between probability bounds and the decision threshold.

Although a decision-threshold policy is less expensive than is the iterative computation of EVC, the threshold analysis has several disadvantages, when compared to EVC metareasoning. A system based on a decision-threshold strategy does not have the ability to halt computation and to take action in cases when reasoning becomes worthless *before* a threshold is reached. Also, such a computer system would be unable to compare the value of a set of alternative reasoning methods and actions. Such value comparisons are useful for choosing among different algorithms, for deciding whether to respond to an observation with an immediate reflex action, and for considering the benefits and risks of committing a large quantity of resource to a noninterruptible problem-solving strategy. Nevertheless, we might wish to compare the value of threshold-based and more sophisticated, EVC-based control.

We shall explore in Section 7.2, the EVM of Protos metareasoning for decision making in several contexts, based on the use of several belief networks. We shall investigate the following forms of EVM:

- EVM($\mathcal{M}^N, \mathcal{M}^C, I, \xi$): The difference between the value of Protos normative-metareasoning techniques ($\mathcal{M}^N$) and the value of solving the entire inference problem ($\mathcal{M}^C$)

- EVM($\mathcal{M}^{p^*}, \mathcal{M}^C, I, \xi$): The difference between the value of acting with the best decision when a decision threshold (p^*) is crossed ($\mathcal{M}^{p^*}$) and the value of solving the entire inference problem

- EVM($\mathcal{M}^N, \mathcal{M}^{p^*}, I, \xi$): The difference between the value of Protos normative metareasoning and a decision-threshold analysis

These measures of EVM depend on the performance of object-level reasoning. We shall explore the EVM for a reasoning system armed with the bounded-conditioning strategy. The values of the EVM measures would change if we endowed Protos' object level with additional inference procedures.

7.1.3 Case Analysis and Summarization

To consider the value of normative metareasoning, we assume as a reference decision the action dictated by instantaneous complete inference. We use the probabilities, computed after an exact analysis, as *gold-standard* probabilities. Expected utilities are assigned to actions, taken after some period of time, by using the exact probabilities ϕ to weight the utility of alternative outcomes. We can use the gold-standard probabilities also to compute a utility of ideal immediate action, by considering the utility of outcomes dictated by the time-independent utility model.

I developed a case summarizer that can be invoked within the Protos system. The facility examines individual Protos decisions, and reports several measures of EVM, in addition to the value of an instantaneous ideal action—the action that would be determined immediately by an infinitely powerful computer. I have used the Protos case summarizer to evaluate the value of Protos reasoning about medical decision problems for several belief networks, including the ALARM network, described in Chapter 4, and the *VentPlan* and *DxNet* networks, which I shall introduce in Sections 7.2.2 and 7.2.3

Although the medical decision problems and patient-specific utility models were designed with the assistance of an emergency-room physician, the examples are intended solely for illustrating the behavior of Protos. I have assumed greater stakes and larger time-dependent losses in the value of outcomes than would be expected in most medical situations. Such time-dependency allows me to demonstrate the concepts of normative metareasoning under bounded resources with our current belief networks. A physician typically has more time for information gathering and analysis than we have assumed in our time-dependent utility models. I discovered that our largest, stable belief networks for time-pressured medical care are not complex enough to exercise the full power of flexible inference and normative metareasoning. I await the development of larger, more expressive belief networks to test these techniques with more realistic time-dependent models.[1] For the cases analyzed, I have assumed time dependencies that are more appropriate for describing the cost of delay for a computer-based reasoning system designed for fast-paced autonomous decision making about such goals as maintaining homeostasis in a trauma patient or coordinating a time-pressured manufacturing process.

7.2 Sample Case Analyses

We shall examine the performance of Protos in several different situations. First, we shall explore normative metareasoning with the use of the ALARM network. Then, we shall investigate decision problems with the VentPlan and DxNet belief networks. We shall explore decision problems that were generated by conditioning the belief network on salient observations that can be explained by two different syndromes. We shall assume, in the cases, that only one syndrome is present and that no new information becomes available during the analysis.

7.2.1 Reasoning with the ALARM Network

Let us first investigate the control of inference in the ALARM network. We consider the case of a binary decision problem involving the time-dependent treatment of a patient who suddenly demonstrates (1) extremely low blood pressure, (2) tachycardia (an extremely fast heart rate), and (3) hypoxemia (low levels of oxygen in the blood). Assume that a physician has ruled out all disorders that might cause these symptoms,

[1]The evolving QMR-DT belief network for internal medicine diagnosis (Shwe et al., 1990a) holds promise as a valuable testbed for Protos' metareasoning techniques.

except for two competing syndromes: *left-ventricular failure* (H_1) and *hypovolemia* (H_2). Hypovolemia is a dangerous state of decreased blood volume (due, for example, to dehydration or bleeding). For a patient with this disorder, the low level of fluids in the circulatory system causes reduced cardiac efficiency and cardiac output. These effects, in turn, lead to low blood pressure, and thus to poor oxygenation of the major organs. As part of a homeostatic reflex, the heart rate increases in an attempt to raise the blood pressure. Left-ventricular failure (LVF) is a serious condition in which the main pumping chamber of the heart is weakened; like hypovolemia, it causes low blood pressure and hypoxia.

Although hypovolemia and LVF share salient symptomology, the treatments for these pathophysiologic states conflict with each other. The treatment for hypovolemia is to give fluids or blood to the patient to restore hydration to a normal level. In contrast, a primary treatment for LVF is to reduce the quantity of liquids in the body, to decrease the pumping burden on a weakened heart, and to reduce the quantity of liquids that filter into the lungs because blood is not being pumped effectively by the heart. Thus, as part of a treatment for LVF, a physician may give a diuretic—a drug that reduces the quantity of fluids in the body. Erroneously treating a patient who has LVF with fluid-replacement therapy, or treating a patient who has hypovolemia with diuretic therapy, can threaten the life of the patient.

Assume that a physician has turned to Protos for assistance with this cardiac decision problem. Protos has been updating its belief about the competing disorders, given observations about the patient, by propagating observations through the ALARM network. We shall now explore feasible patient utility models and examples of belief-network inference.

Table 7.1 displays a Protos time-dependent utility model. The model describes the value of the four possible outcomes, and contains information about the time-dependent nature of the quality of treatment for patients who manifest symptomology that can be explained by LVF or hypovolemia. In this case, the time-dependency values are decay constants for a negative exponential function. The decay constants indicate that, in this context, the time criticality for treating LVF is greater than the criticality for treating hypovolemia.

Outcomes		U_{t_α}	Time Dependence		Convergence
Treat Hypovolemia	Hypovolemia	1.00	Exp	0.0001	u(LVF,Hyp)
Treat LVF	Hypovolemia	0.40	Ø	–	0.40
Treat LVF	LVF	0.20	Exp	0.001	u(Hyp,LVF)
Treat Hypovolemia	LVF	0.05	Ø	–	0.05

Table 7.1: This patient-specific utility model represents sample information about the time-dependent nature of the quality of treatment for patient who manifests symptomology that can be explained by left-ventricular failure (LVF) or hypovolemia (Hyp). In this model, the cost of delay in treating LVF is described by an exponential decay constant that is 10 times larger than the constant used to describe the cost of delay in treating hypovolemia.

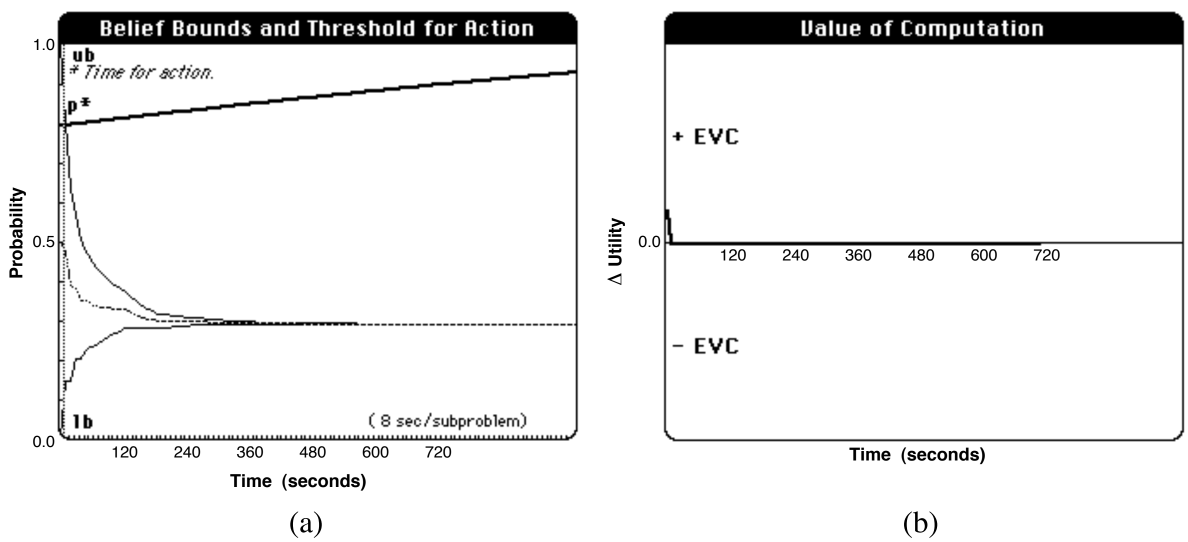


Figure 7.1: Time-dependent inference and ideal action.
The graph in (a) shows the convergence of upper and lower bounds on the probability of LVF with computation for a new update. The graph in (b) displays the EVC/BC for this evidential update. The graph juxtaposes the EVC/BC for the time-dependent problem (dark curve) with the time-independent EVC/BC (lighter curve). The EVC/BC diminishes to 0 as the decision threshold is crossed.

Let us now examine the ideal control of a belief-update procedure. Figure 7.1(a) shows the update of belief in LVF versus hypovolemia, given an observation that the *central venous pressure* (CVP) is *high*. Here, Protos was previously informed that the patient displayed *low cardiac output* and had *no previous history of LVF*. Figure 7.1(a) also displays the mean value between the bounds and the time-dependent decision threshold. For this update, the system requires 8 seconds to solve each subproblem. It must solve 108 subproblems to generate an exact probability. A vertical line indicates Protos' decision to halt and to recommend action after only 9 seconds of reasoning. At this time, one subproblem has been solved. In this case, the cessation of computation has occurred at a decision threshold. Figure 7.1(b) displays the EVC/BC for this update. Note that the value of the EVC becomes 0 as the decision threshold is crossed.

Figure 7.2 displays a graphical analysis of the utility of treating for LVF and for hypovolemia as a function of the probability of LVF. The graph displays the upper and lower bounds (ub, lb), the mean value between these bounds (mean), and the decision threshold, p^*, at the time Protos recommended treatment for hypovolemia. The best action is dictated by the position of the mean. In this analysis, the mean is below p^*. The graph also displays the final probability (p) that would be computed with the complete solution of the inference problem. The position of the exact probability demonstrates that an instantaneous analysis also would have indicated that the ideal action is to treat for hypovolemia.

The summary generated by Protos' EVM analyzer is displayed in Figure 7.3. The summary describes the value of ideal action computed by an infinitely fast computer. In this situation, immediate action, based on perfect computation, would have been worth 0.72. Next, the summary compares the value of a complete analysis with the value of Protos' recommendation. The complete analysis (solving all 108 subproblems) would have required a delay of 865 seconds. The action, recommended after that analysis (to treat for hypovolemia), would have been worth 0.66. Protos' decision—to treat for hypovolemia after waiting just 9 seconds—is worth 0.72. This is the same value (at this level of precision) as the instantaneous analysis. The value of Protos' action is 0.06 greater than the value of a complete analysis.

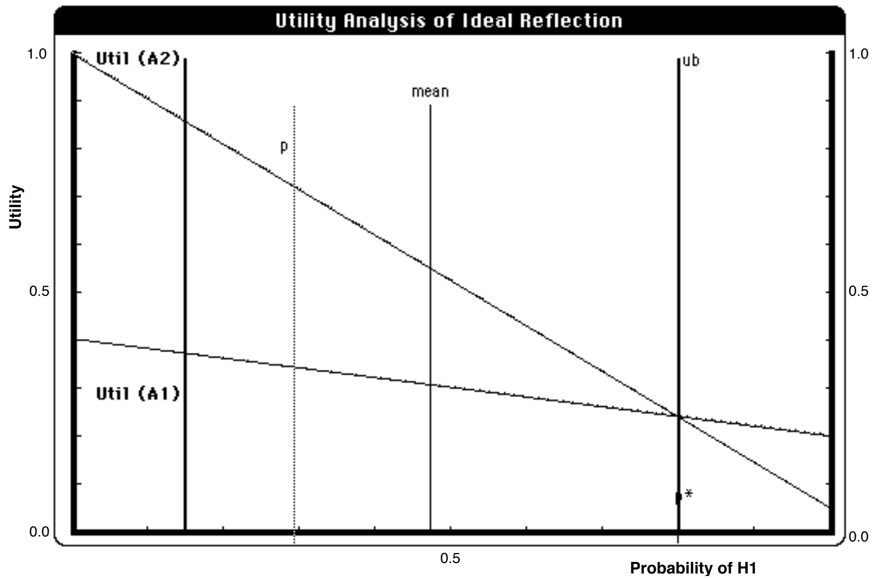
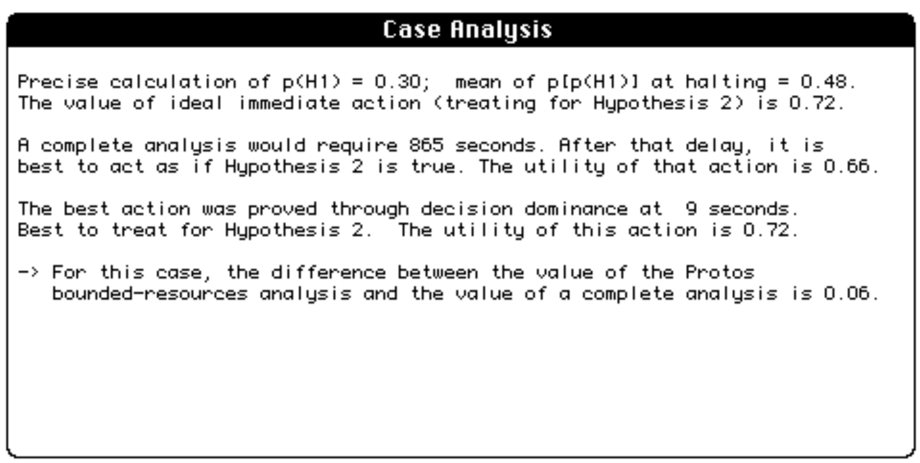


Figure 7.2: Utility analysis of decision and time for action recommended by Protos. This graph displays the utility of treating for LVF [Util(A_1)] and for hypovolemia as a function of the probability of LVF (H_1). The graph displays the upper and lower bounds (ub, lb), the mean value between these bounds (mean), the actual probability (p), and the decision threshold p^* at the time Protos recommended treatment. The bounded and instantaneous perfect analyses both indicate that the ideal action is to treat for hypovolemia.


Case Analysis

Precise calculation of p(H1) = 0.30; mean of p[p(H1)] at halting = 0.48.
The value of ideal immediate action (treating for Hypothesis 2) is 0.72.

A complete analysis would require 865 seconds. After that delay, it is
best to act as if Hypothesis 2 is true. The utility of that action is 0.66.

The best action was proved through decision dominance at 9 seconds.
Best to treat for Hypothesis 2. The utility of this action is 0.72.

-> For this case, the difference between the value of the Protos
bounded-resources analysis and the value of a complete analysis is 0.06.

Figure 7.3: Case analysis.

This is text generated by an evaluation feature of Protos summarizing key aspects of Protos' decision. The analysis compares the value of a complete analysis with the value of Protos' recommendation. For this case, Protos' reasoning is worth 0.06 more than a complete solution.

Outcomes		U_{t_α}	Time Dependence		Convergence
Treat Hypovolemia	Hypovolemia	1.00	Exp	0.0002	u(LVF,Hyp)
Treat LVF	Hypovolemia	0.40	Ø	–	0.40
Treat LVF	LVF	0.20	Exp	0.001	u(Hyp,LVF)
Treat Hypovolemia	LVF	0.05	Ø	–	0.05

Table 7.2: Another utility model for the cardiac decision problem. Here, the decay constant that describes the cost of delay in treating hypovolemia has been doubled.

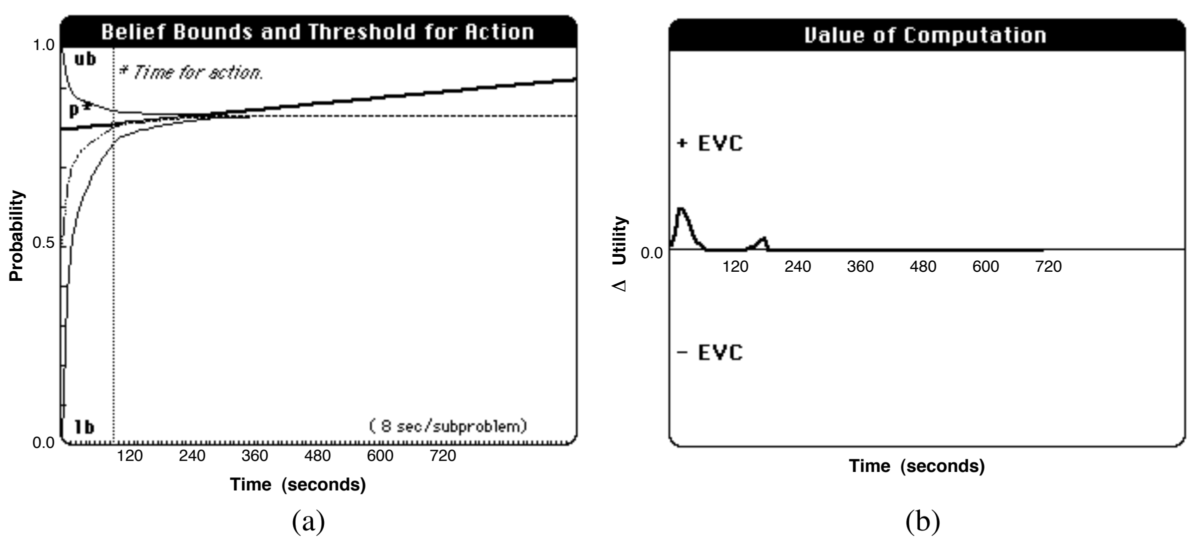


Figure 7.4: Time-dependent inference and ideal action.
(a) Convergence of upper and lower bounds on the probability of left-ventricular failure with computation. (b) The EVC/BC for this evidential update.

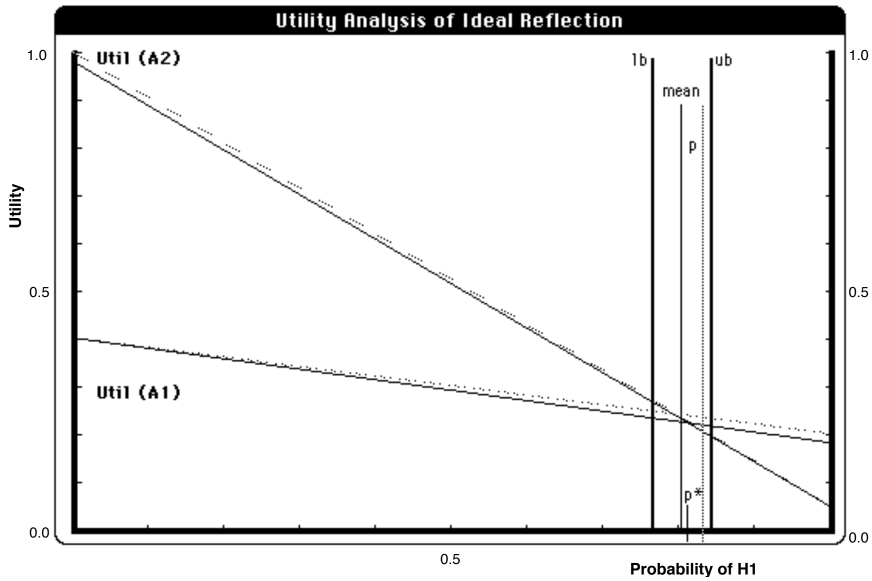


Figure 7.5: Utility analysis of decision and time for action recommended by Protos. This graph displays the utility of treating for LVF [Util(A_1)] and for hypovolemia [Util(A_2)] as a function of the probability of LVF (H_1).

Case Analysis

```
Precise calculation of p(H1) = 0.83;  mean of p[p(H1)] at halting = 0.81.
The value of ideal immediate action (treating for Hypothesis 1) is 0.23.

A complete analysis would require 865 seconds. After that delay, it is
best to act as if Hypothesis 2 is true. The utility of that action is 0.18.

Best to halt in 89 seconds and treat as if Hypothesis 2.
This action has a utility of 0.21.

-> For this case, the difference between the value of the Protos
   bounded-resources analysis and the value of a complete analysis is 0.02.

 * A policy of halting at a decision threshold would dictate action after
      241 seconds to treat for Hypothesis 2 with a utility of 0.20.

 * EVC metareasoning is 0.00 more valuable than stopping at a threshold.
  A decision threshold policy is 0.02 more valuable than a complete analysis.
```

Figure 7.6: Case analysis.

The analysis compares the value of a complete analysis with the value of Protos' recommendation. Here, Protos also compares the bounded action policy with the value of halting at the decision threshold.

Let us now examine another case. Table 7.2 displays a representation of another utility model for the same decision problem. This utility model indicates that delay in treating hypovolemia is more costly than a delay in treating a patient described by the utility model presented in Table 7.1.

Figure 7.4(a) shows a different update of belief in LVF. The graph shows a trace of Protos' inference about relationships in the ALARM belief network. Here, Protos is considering a new observation that the *CVP is normal*. Protos was previously informed that the patient displayed *low stroke volume* and had a *previous history of LVF*. Figure 7.4(a) displays the mean value between the bounds and the time-dependent decision threshold. Here, the vertical line indicates Protos' decision to halt in 89 seconds after solving 11 out of 108 total subproblems. Protos recommends that the patient should be treated for hypovolemia. Figure 7.4(b) displays the EVC/BC over time. The cessation of computation has occurred *before* a decision threshold has been crossed because the EVC/BC diminishes to zero before a bound reached that threshold. Note that, in this case, the value of the EVC/BC becomes positive again at 120 seconds. Protos' lookahead machinery does not detect this brief positive swing in the value of EVC/BC; the second, smaller maxima occurs beyond the horizon of the system's myopic analysis.

Figure 7.5 displays a graphical analysis of the utility of treating for LVF and for hypovolemia as a function of the probability of LVF. In this case, the delay before action has had a significant effect on the value of time-dependent outcomes. The broken lines adjacent to the solid utility lines indicate the utility of the two actions before the time-dependent decay of the value of the outcomes. The graph also displays the upper and lower bounds (ub, lb), the mean value between these bounds (mean), and the decision threshold p^* at the time Protos recommended treatment for hypovolemia. The best action is dictated by the position of the mean. In the analysis, the mean is below the revised p^*. The graph also displays the final probability (p) that would be computed with the solution of the entire inference problem. For this example, an instantaneous analysis would have indicated that it is ideal to treat for LVF.

The summary generated for this case by Protos' EVM analyzer is displayed in Figure 7.6. As indicated in the summary, a perfect instant analysis would have been

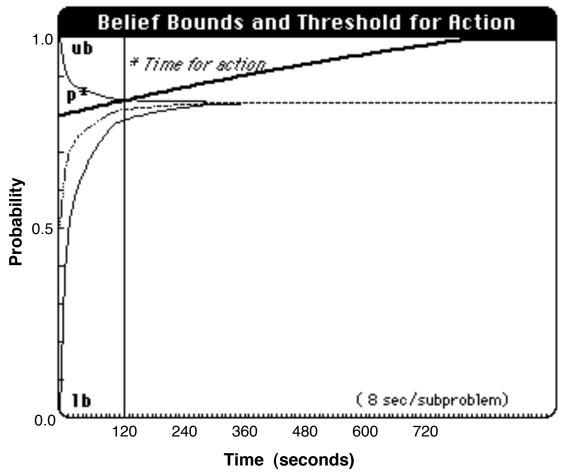
Outcomes		U_{t_α}	Time Dependence		Convergence
Treat Hypovolemia	Hypovolemia	1.00	Exp	0.0002	u(LVF,Hyp)
Treat LVF	Hypovolemia	0.40	Ø	–	0.40
Treat LVF	LVF	0.20	Exp	0.001 → **0.002**	u(Hyp,LVF)
Treat Hypovolemia	LVF	0.05	Ø	–	0.05

Table 7.3: Increasing the time-criticality for LVF. In this model, we double the exponential decay constant that describes the losses with time for the outcome of treating for LVF in the presence of LVF.

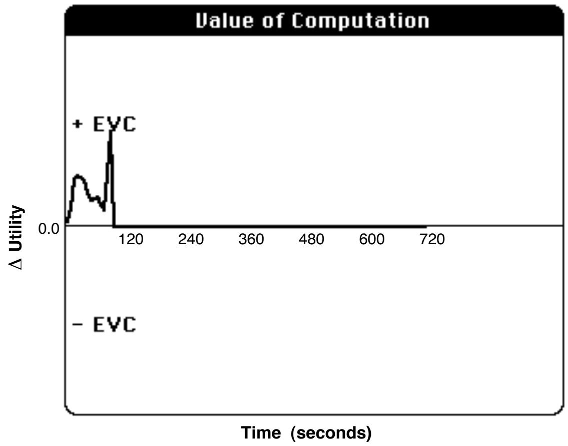
worth 0.23. Protos' decision, to treat for hypovolemia after waiting for 89 seconds, is worth 0.21—0.02 less than value of an instantaneous analysis, but 0.03 more than the 0.18 value of complete analysis (solving all 108 subproblems). In this case, Protos ceased computation and acted before the decision threshold. Thus, Protos also can compare the value of its decision with the value of a decision based on a decision-threshold policy. Protos' action is worth 0.01 more than the value of a threshold-based analysis.

Let us now explore the sensitivity of Protos' behavior to small changes in the time-dependencies represented in the cardiac utility model. As indicated in Table 7.3, we shall now double the exponential decay constant that describes the loss of value for the outcome of treating for LVF when LVF is indeed present. Figure 7.7(a) shows the same convergence of upper and lower bounds on the probability of LVF as described in Figure 7.4. However, with the use of the revised time-dependent utility model, Protos recommends that the patient should be treated for LVF. This recommendation is made when the upper bound crosses the decision threshold after 113 seconds of computation. Figure 7.7(a) shows how the revised stopping time is explained by a more rapid increase of the probability threshold with delay. The graph in Figure 7.7(b) displays the EVC/BC for the revised utility model. Note the difference in the structure of the original and new EVC/BC.

Figure 7.8 displays a graphical utility analysis of the revised state of affairs at the time of action. Note that the bounds are tighter than in they were at halting time in the previous analysis because more computation time has been expended. Also,



(a)



(b)

Figure 7.7: Time-dependent inference and ideal action for the new utility model. (a) The same convergence of upper and lower bounds on the probability of LVF as described in Figure 7.4. Protos now recommends action at a decision threshold. (b) The EVC/BC for the new utility model.

the decision threshold p^* is higher than it was at the time of action in the original analysis.

The case-analysis summary generated by Protos' EVM analyzer for this case is displayed in Figure 7.9. As indicated in the summary, a perfect immediate analysis would have been worth 0.23. Protos' decision, to treat for LVF, after waiting for 113 seconds, is worth 0.20—that is 0.03 less than an instantaneous complete analysis, but 0.02 more than an actual complete analysis.

7.2.2 Reasoning with the VentPlan Network

We shall now make use of a different belief network to investigate another decision problem with Protos. We shall use the VentPlan network (Rutledge et al., 1989). This 31-node belief network is displayed in Figure 7.10. VentPlan was designed for the representation of the change in the validity of reported observations about a patient's physiology over time. The network has a smaller cutset and number of instances than does ALARM: three cutset nodes yield 24 inference subproblems for

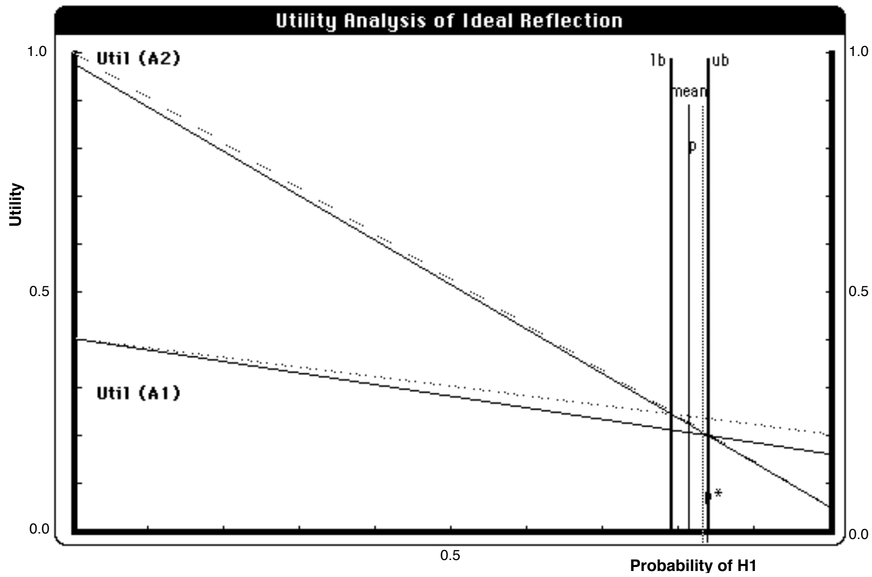
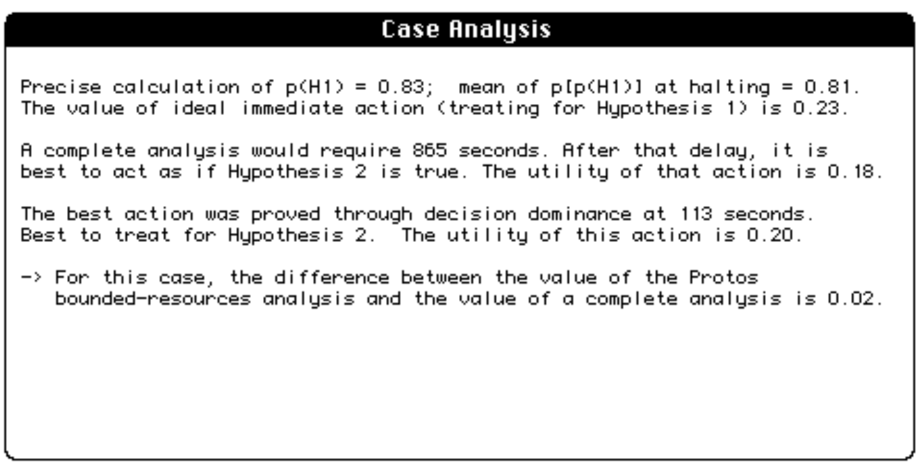


Figure 7.8: Utility analysis of action.

This graph displays the utility (crossing solid lines) of treating for LVF [Util(A_1)] and for hypovolemia (action A_2) as a function of the probability of LVF (H_1). The graph also shows the initial utilities (crossing broken lines), the upper and lower bounds (ub, lb), the mean value between these bounds (mean), the decision threshold p^*, and the final probability (p).



Case Analysis

Precise calculation of p(H1) = 0.83; mean of p[p(H1)] at halting = 0.81.
The value of ideal immediate action (treating for Hypothesis 1) is 0.23.

A complete analysis would require 865 seconds. After that delay, it is best to act as if Hypothesis 2 is true. The utility of that action is 0.18.

The best action was proved through decision dominance at 113 seconds.
Best to treat for Hypothesis 2. The utility of this action is 0.20.

-> For this case, the difference between the value of the Protos bounded-resources analysis and the value of a complete analysis is 0.02.

Figure 7.9: Case analysis.

This computer-generated summary tells us that, in this situation, Protos' reasoning is worth 0.02 more than a complete analysis.

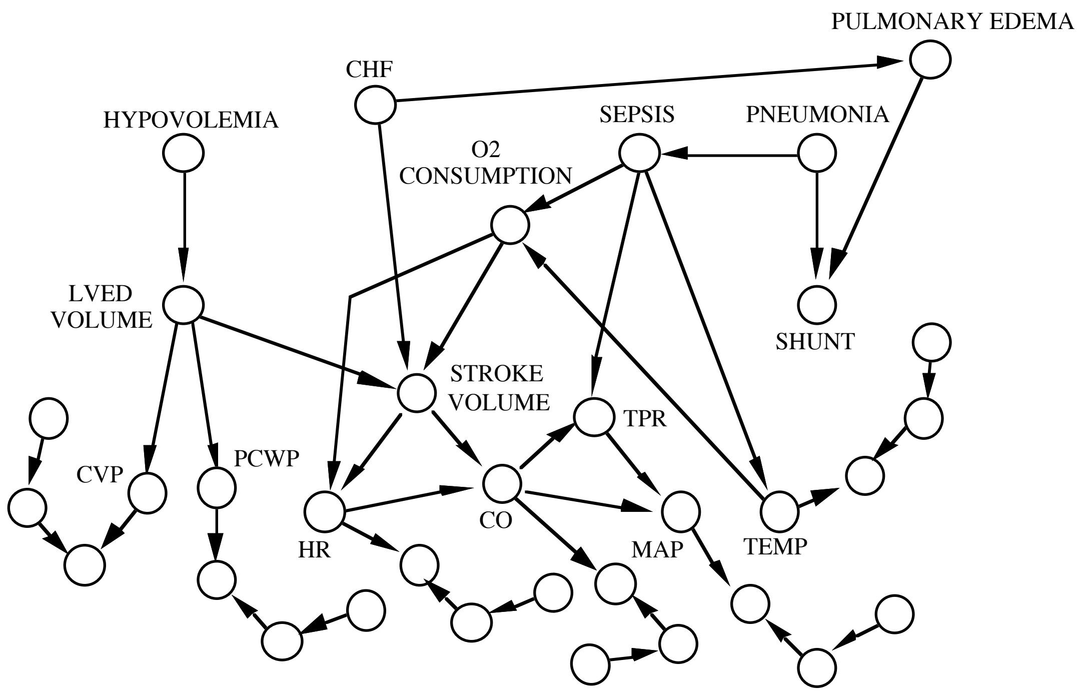


Figure 7.10: The VentPlan belief network.
This 31-node network is designed for reasoning about the validity of patient information over time. Unlabeled nodes represent distinctions about error in the reported information over time. (See Appendix B for a legend to abbreviations in this belief network.)

consideration by bounded-conditioning.

We shall consider the case of a binary decision problem involving the time-dependent treatment of a patient who suddenly shows (1) severe shortness of breath, (2) a productive cough, and (3) signs of hypoxemia. We assume that a physician has ruled out all disorders that might cause these symptoms, except for two competing syndromes: *pneumonia* (H_1) and *congestive heart failure* or *CHF* (H_2). Table 7.5 displays a representation of a utility model for this decision problem. In this case, the model indicates that delay in treating CHF incurs greater losses than does a delay in treating pneumonia.

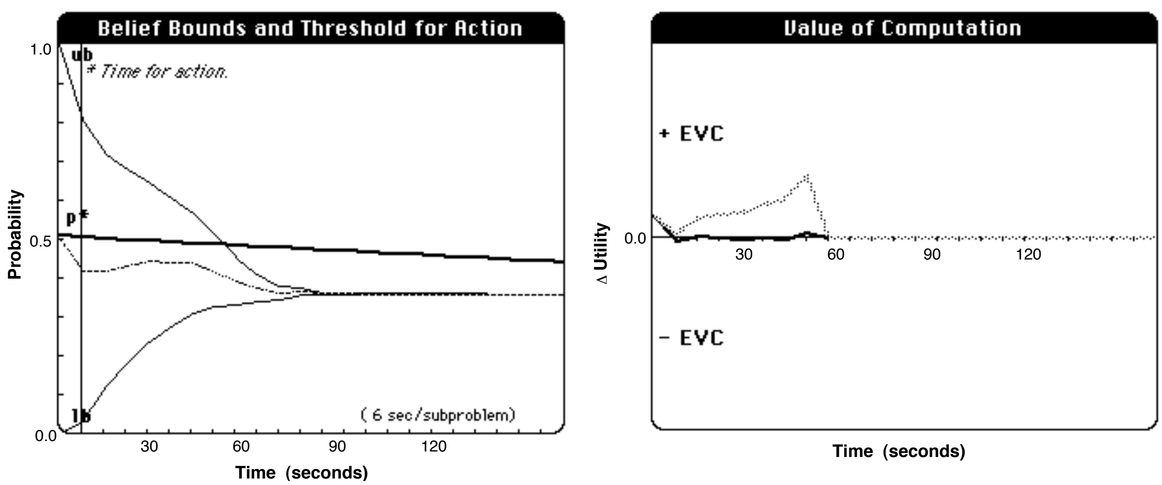
Figure 7.11(a) shows an evidential update for the probability of pneumonia versus CHF. In this case, Protos bounds the probability of pneumonia, given an observation that the patient has a *high fever*. Protos was previously informed that the patient shows a *very high respiratory shunt* and has a *high mean arterial pressure* (MAP). Figure 7.11(a) displays the mean value between the bounds and the time-dependent decision threshold. The vertical line indicates Protos' decision to halt after solving 1 subproblem. In this case, Protos has recommended that action be taken before a decision threshold has been crossed. The graph in (b) displays the EVC/BC (darker line) for this evidential update. The graph contrasts the EVC/BC for the time-dependent problem with the time-independent EVC/BC (lighter line). Analyzing the graph of the EVC/BC can give us insight about why Protos recommended action before reaching a decision threshold: The EVC/BC becomes nonpositive before the threshold is reached.

Figure 7.12 displays the utility of treating for CHF [Util(A_2)] and for pneumonia [Util(A_1)] as a function of the probability of pneumonia (H_1) for the utility model in Table 7.4. The best action, as indicated by the position of the mean, with respect to p^*, is to treat for CHF. In this case, the final probability (p) is close to the mean at the time computation was halted; for this example, an instantaneous analysis also would have indicated that the ideal action is to treat for CHF.

The summary generated for this case by Protos' EVM analyzer is displayed in Figure 7.13. The value of Protos' bounded decision is nearly the same as the value of a perfect, instantaneous analysis. The analysis compares the value of a complete

Outcomes		U_{t_α}	Time Dependence		Convergence
Treat CHF	CHF	0.35	Exp	0.002	u(Pneu,CHF)
Treat Pneumonia	CHF	0.03	Ø	–	0.03
Treat Pneumonia	Pneumonia	0.75	Exp	0.0001	u(CHF,Pneu)
Treat CHF	Pneumonia	0.45	Ø	–	0.45

Table 7.4: Patient-specific utility information for a respiratory decision problem. This model represents sample information about the time-dependent nature of the quality of treatment for patient who manifests symptomology that can be explained by CHF or by pneumonia (Pneu).



(a) (b)

Figure 7.11: Time-dependent inference and ideal action.
(a) Convergence of upper and lower bounds on the probability of pneumonia with computation. (b) The EVC/BC for this evidential update. Notice that Protos has recommended action before a decision threshold has been crossed, because the EVC/BC becomes small before the threshold is reached. The graph contrasts the EVC/BC for the time-dependent problem with the time-independent EVC/BC (lighter).

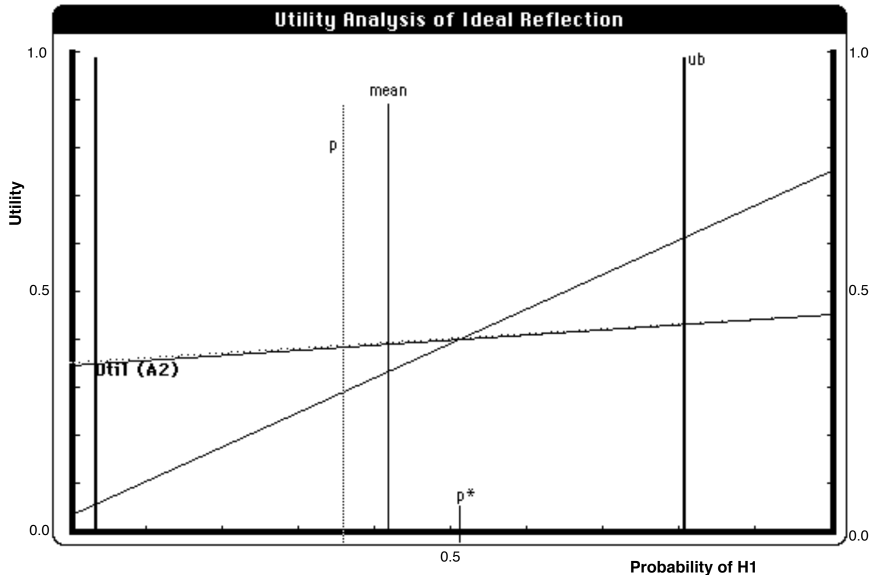


Figure 7.12: Utility analysis of decision and time for action recommended by Protos. This graph displays the utility of treating for CHF [Util(A_2)] and for pneumonia [Util(A_1)] as a function of the probability of pneumonia (H_1) for the utility model under consideration. The best action, as indicated by the position of the mean, is to treat for CHF.

analysis, and the value of a threshold analysis with the value of Protos' recommendation. In this case, Protos' reasoning is worth 0.06 more than a complete analysis, and 0.04 more than a threshold-based analysis.

Let us now explore the ideal action indicated by the same update with a different utility model. Table 7.5 displays another time-dependent utility model that represents the preferences of a patient who manifests symptomology that can be explained by either CHF or pneumonia.

The graph in Figure 7.14(a) shows the convergence of bounds on the probability of pneumonia with computation. A new decision threshold, generated from the time-dependent utilities in the utility model, is displayed. In this case, Protos halts computation and recommends action after solving 4 subproblems. At this time, the lower bound crosses over the decision threshold. Figure 7.14(b) displays the curves for the time-dependent EVC/BC (darker line) and time-independent (lighter line) decision problems. The time-dependent EVC/BC becomes 0 as the bound crosses the threshold probability.

Figure 7.15 displays the utility of treating for CHF [Util(A_2)] and for pneumonia

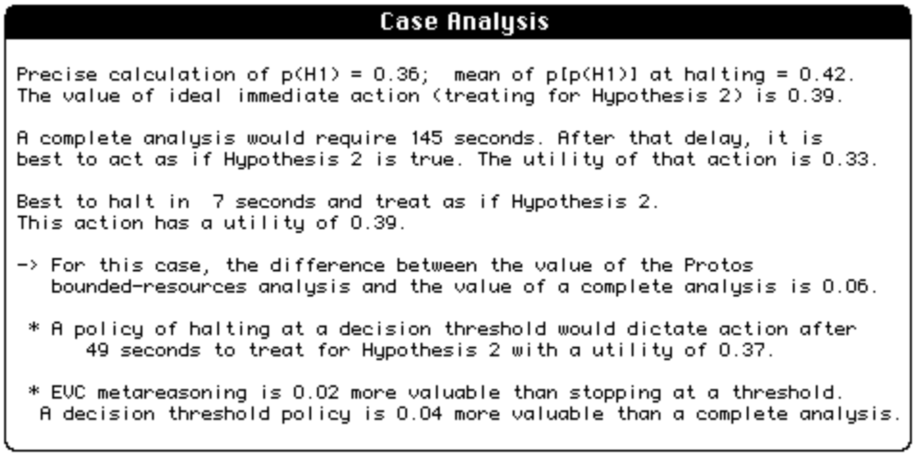
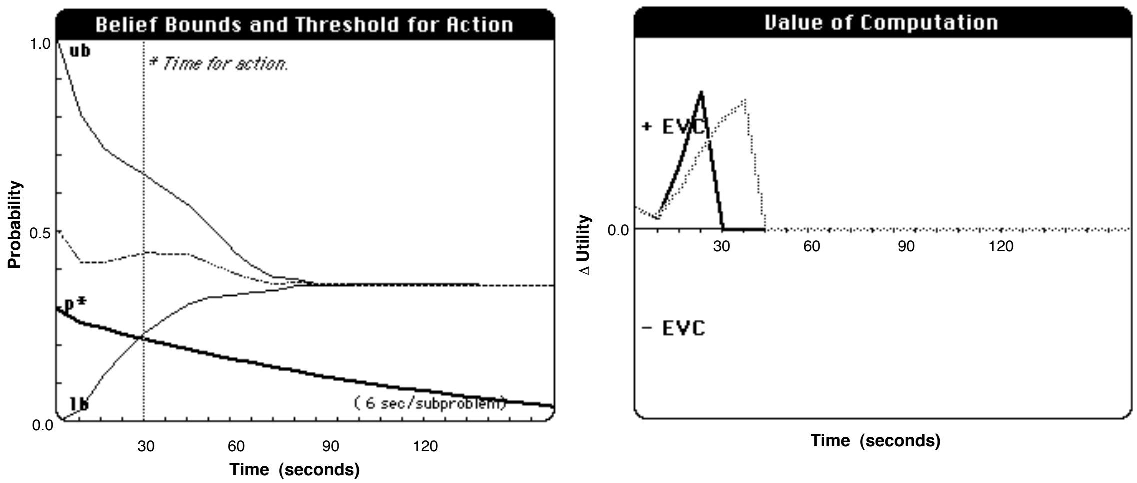


Figure 7.13: Case analysis.
This summary analysis compares the value of a complete analysis, and the value of a threshold analysis with the value of Protos' recommendation. In this case, Protos' reasoning is worth 0.04 more than a threshold-based analysis.

Outcomes		U_{t_α}	Time Dependence		Convergence
Treat CHF	CHF	**0.20**	Exp	**0.015**	u(Pneu,CHF)
Treat Pneumonia	CHF	**0.01**	Ø	–	0.01
Treat Pneumonia	Pneumonia	**0.85**	Exp	**0.0015**	u(CHF,Pneu)
Treat CHF	Pneumonia	**0.40**	Ø	–	0.40

Table 7.5: Another time-dependent utility model. This model represents new preferences of a patient who manifests symptomology that can be explained by CHF or pneumonia (Pneu).



(a) (b)

Figure 7.14: Time-dependent inference and ideal action.
The graph in (a) shows convergence of bounds on the probability of pneumonia with computation, and the time-dependent decision threshold. The graph in (b) displays the curves for the time-dependent EVC/BC (darker line) and time-independent (lighter line) decision problems.

[Util(A_1)] as a function of the probability of pneumonia (H_1) for the utility model in Figure 7.5. The best action, as indicated by the position of the lower bound on the probability of pneumonia (H_1), with respect to p^*, is to treat for pneumonia.

The summary generated for this case by Protos' EVM analyzer is displayed in Figure 7.16. As indicated in the summary, a perfect instant analysis would have been worth 0.31. Protos' decision, to treat for hypovolemia, made after 25 seconds, is worth 0.30. The utility of this decision is 0.01 less than the value of the instantaneous analysis, but is 0.05 more than the value of complete analysis (solving all 24 subproblems).

Table 7.6 shows another utility model for the same VentPlan decision problem. The implications of this model are displayed in Figure 7.17(a). The graph shows that the time-dependent decision threshold remains within the envelope, close to the mean of the bounds, until the problem is near completion. At this time, the lower bound on the probability passes through the decision threshold, indicating that the best action is to treat for pneumonia. Notice that the inference problem is almost completely solved prior to the decision threshold being crossed. The graph in Figure

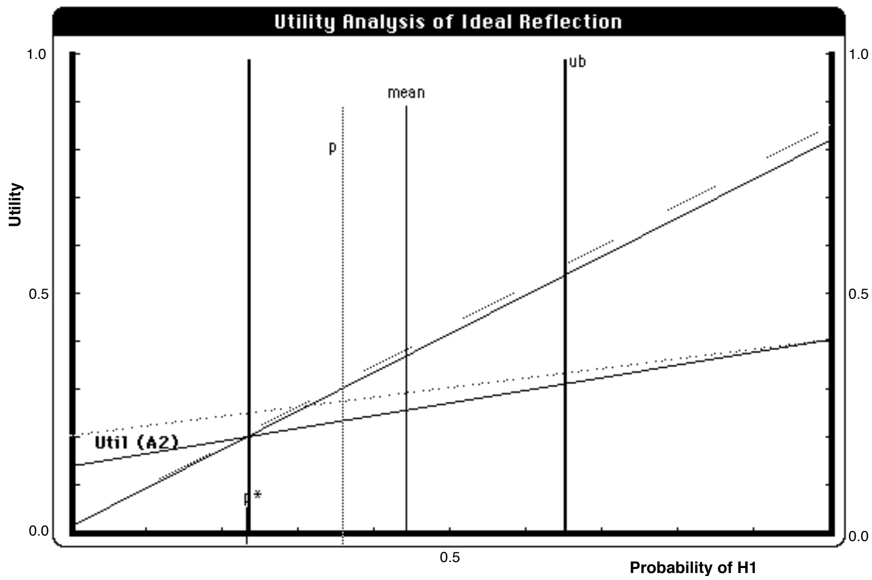


Figure 7.15: Utility analysis of decision and time for action recommended by Protos. This graph displays the utility of treating for CHF [Util(A_2)] and for pneumonia [Util(A_1)] as a function of the probability of pneumonia (H_1) for the utility model under consideration. The broken lines adjacent to the solid utility lines indicate the utility of the two actions before the time-dependent decay of the utility of the outcomes. The best action is to treat for pneumonia.

Case Analysis

Precise calculation of p(H1) = 0.36; mean of p[p(H1)] at halting = 0.43.
The value of ideal immediate action (treating for Hypothesis 1) is 0.31.

A complete analysis would require 145 seconds. After that delay, it is
best to act as if Hypothesis 1 is true. The utility of that action is 0.25.

The best action was proved through decision dominance at 25 seconds.
Best to treat for Hypothesis 1. The utility of this action is 0.30.

-> For this case, the difference between the value of the Protos
bounded-resources analysis and the value of a complete analysis is 0.05.

Figure 7.16: Case analysis.

The analysis of Protos' decision compares the value of a complete analysis, and the value of a threshold analysis, with the value of Protos' recommendation. In this case, Protos' reasoning is worth 0.05 more than a complete analysis.

Outcomes		U_{t_α}	Time Dependence		Convergence
Treat CHF	CHF	**0.25**	Exp	**0.0008**	u(Pneu,CHF)
Treat Pneumonia	CHF	**0.03**	∅	–	0.03
Treat Pneumonia	Pneumonia	**0.65**	Exp	**0.0002**	u(CHF,Pneu)
Treat CHF	Pneumonia	**0.25**	∅	–	0.25

Table 7.6: This is a different utility model for treating a patient who manifests symptomology that can be explained by either CHF or pneumonia (Pneu).

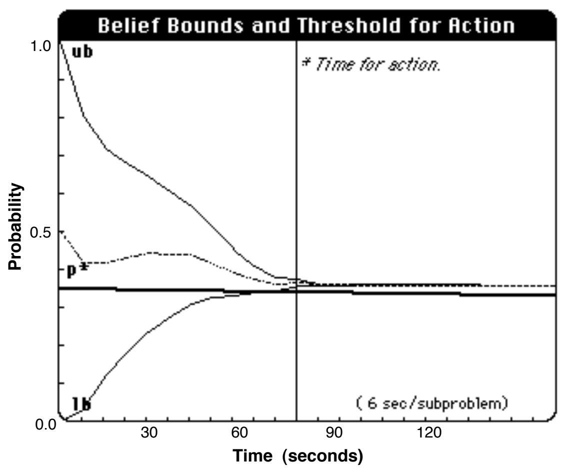
7.17(b) displays the curves for the time-independent and time-dependent EVC/BC. For this utility model, these measures of EVC are similar. This graphical analysis in Figure 7.18 shows that, for the utility model begin considered, the final belief is near the decision threshold. As revealed in the analysis summary displayed in Figure 7.19, the values of action determined by an instantaneous analysis, by complete inference, and by Protos' bounded-resource recommendation are all 0.25. In this case, we gain nothing by using Protos' metareasoning.

7.2.3 Reasoning with the DxNet Network

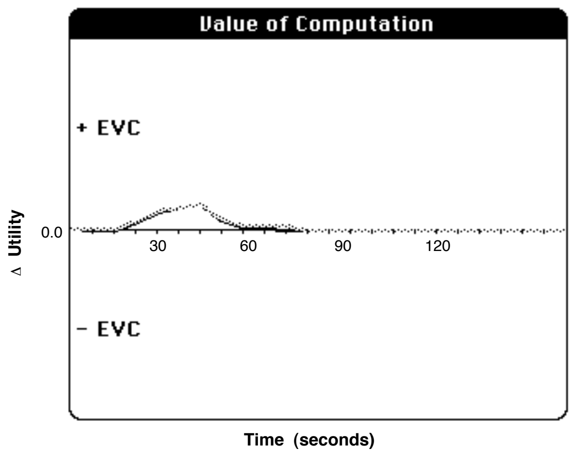
We shall now investigate Protos' decisions under time pressure for a decision problem that draws on relationships represented in DxNet. DxNet's belief network is displayed in Figure 7.20. The network consists of 81 nodes. The network was created to represent the unreliability of reported information, and thus has sets of nodes for representing error in observations. The network can be decomposed into a set of 54 subproblems with a cutset of 4 nodes.

We consider assisting a physician in treating a patient who displays symptomology that can be explained by a pulmonary embolism (H_1) or by pneumonia (H_2). Table 7.7 displays a utility model for this decision problem. In this case, the model indicates that delay in treating an embolism incurs a greater loss than does a delay in treating pneumonia.

Figure 7.21(a) shows an evidential update for the probability of a pulmonary embolism versus pneumonia. Here, Protos is bounding the probability of an embolism, given an observation that the patient has a *high mean pulmonary arterial pressure*



(a)



(b)

Figure 7.17: Time-dependent inference and ideal action.

The graph in (a) shows convergence of bounds on the probability of pneumonia with computation, and the time-dependent decision threshold. The graph in (b) displays the curves for the time independent and time-dependent EVC/BC. For this utility model, the two measures of EVC are similar.

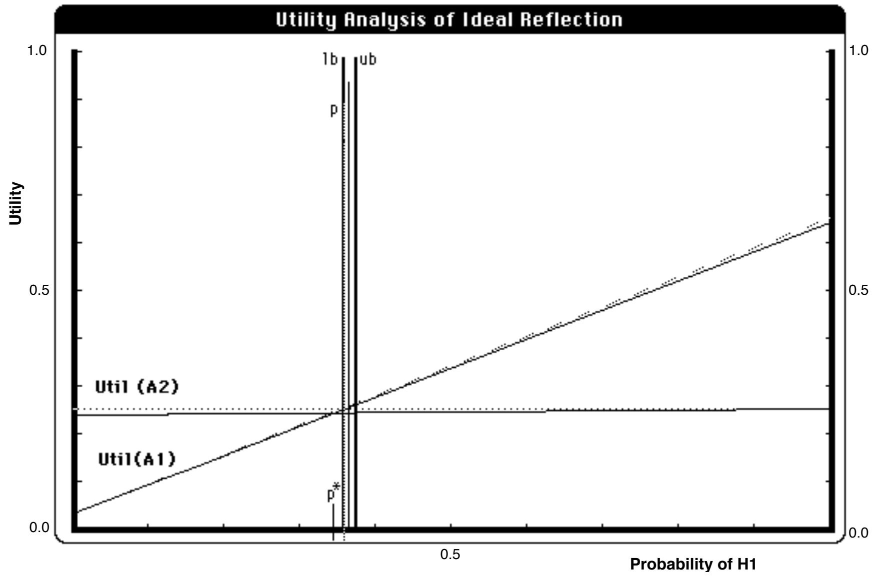


Figure 7.18: Utility analysis of decision and time for action recommended by Protos. This graph displays the utility of treating for CHF [Util(A_2)] and for pneumonia [Util(A_1)] as a function of the probability of pneumonia (H_1) for the utility model under consideration. The best action is to treat for pneumonia. In this case, the inference problem is almost completely solved at the time inference was halted.

Case Analysis

```
Precise calculation of p(H1) = 0.36;  mean of p[p(H1)] at halting = 0.36.
The value of ideal immediate action (treating for Hypothesis 1) is 0.25.

A complete analysis would require 145 seconds. After that delay, it is
best to act as if Hypothesis 1 is true. The utility of that action is 0.25.

The best action was proved through decision dominance at 67 seconds.
Best to treat for Hypothesis 1.  The utility of this action is 0.25.

-> For this case, the difference between the value of the Protos
   bounded-resources analysis and the value of a complete analysis is 0.00.
```

Figure 7.19: Case analysis.

Analysis reveals that, for the utility model and evidential update under consideration, nothing is gained by Protos' analysis.

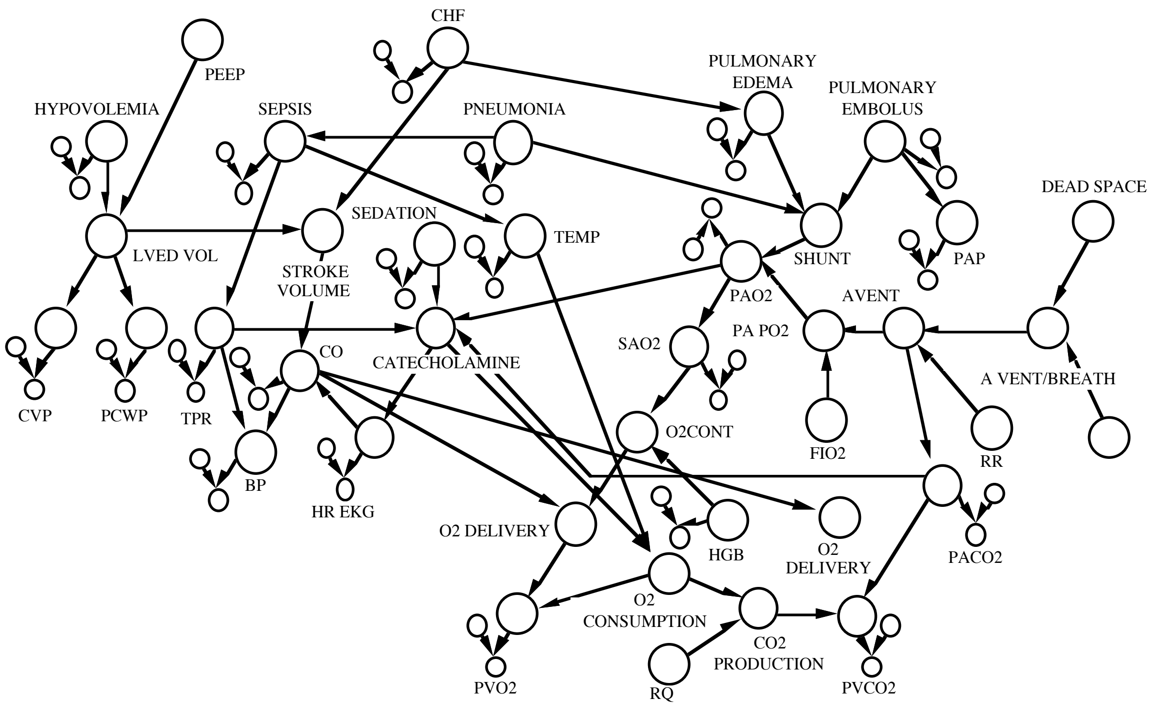


Figure 7.20: The DxNet belief network.
This belief network, created to reason about uncertainty in reported information for ICU medicine, has 81 nodes. (See Appendix B for a legend to abbreviations in this belief network.) The smaller nodes represent variables used to model errors in the measurement of the value of evidence variables.

Outcomes		U_{t_α}	Time Dependence		Convergence
Treat Pneumonia	Pneumonia	1.00	Exp	0.008	u(Embol,Pneu)
Treat Embolism	Pneumonia	0.35	Ø	–	0.35
Treat Embolism	Embolism	0.15	Exp	0.02	u(Pneu,Embol)
Treat Pneumonia	Embolism	0.02	Ø	–	0.02

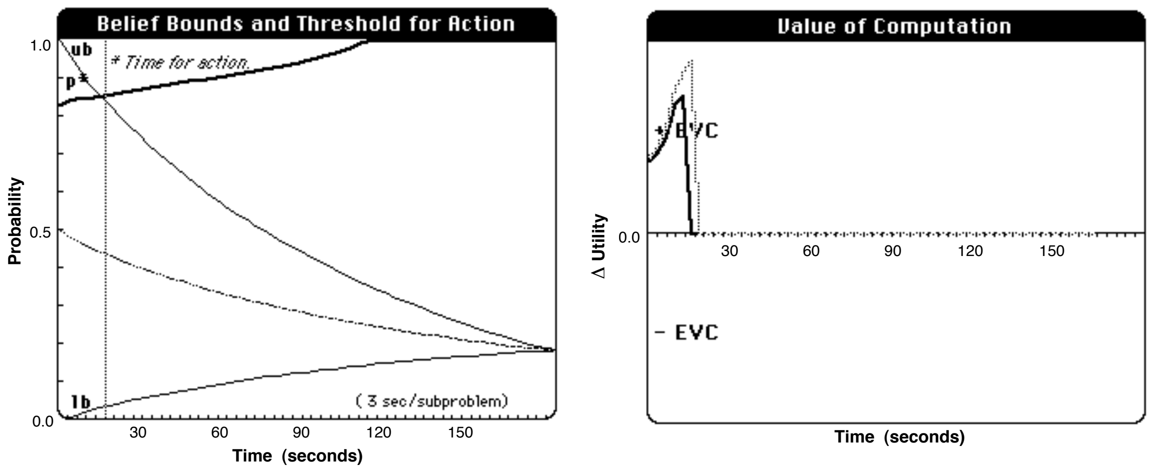
Table 7.7: A utility model for a pulmonary decision problem. This model represents sample information about the time-dependent nature of the quality of treatment for patient who manifests symptomology that can be explained by a pulmonary embolism (Embol) or pneumonia (Pneu).

(PAP). Protos was previously informed that the patient shows a *very high respiratory shunt* and has a *normal blood pressure*.

Figure 7.21(a) displays the mean value between the bounds and the time-dependent decision threshold. The vertical line indicates Protos' decision to halt after solving 6 subproblems. At this point, a decision threshold is reached. The graph in Figure 7.21(b) displays the EVC/BC (darker line) for this evidential update. The graph also displays the time-independent EVC/BC (lighter line).

Figure 7.22 displays the utility of treating for pulmonary embolism [Util(A_1)] and for pneumonia [Util(A_2)] as a function of the probability of embolism (H_1) for the utility model displayed in Table 7.7. The best action, as indicated by the position of the upper bound on the probability of pulmonary embolism with respect to p^*, is to treat for pneumonia. The analysis summary, in Figure 7.23 indicates that Protos' threshold analysis is worth 0.51 more than a complete analysis, and only 0.08 less than instantaneous complete computation.

Let us consider another utility model for the embolism–pneumonia decision problem displayed in Table 7.8. The differences between the time-dependent utilities in this model and those in the model assumed in the previous analysis (displayed in Table 7.7) are highlighted with arrows. The graph in Figure 7.24(a) shows the same convergence of computed bounds on the probability of pulmonary embolism, and a revised time-dependent utility threshold, implied by the new model. The graph in Figure 7.24(b) displays the time-dependent (darker curve) and time-independent EVC/BC



(a) (b)

Figure 7.21: Time-dependent inference and ideal action.
(a) Convergence of upper and lower bounds on the probability of pulmonary embolism with computation. (b) The EVC/BC for this evidential update.

Outcomes		U_{t_α}	Time Dependence		Convergence
Treat Pneumonia	Pneumonia	1.00	Exp	$0.008 \rightarrow \mathbf{0.009}$	u(Embol,Pneu)
Treat Embolism	Pneumonia	$0.35 \rightarrow \mathbf{0.55}$	Ø	–	0.35
Treat Embolism	Embolism	$0.15 \rightarrow \mathbf{0.25}$	Exp	0.02	u(Pneu,Embol)
Treat Pneumonia	Embolism	0.02	Ø	–	0.02

Table 7.8: This model represents a different set of preferences for making a decision about a patient who manifests symptomology that can be explained by pulmonary emboli or pneumonia (Pneu).

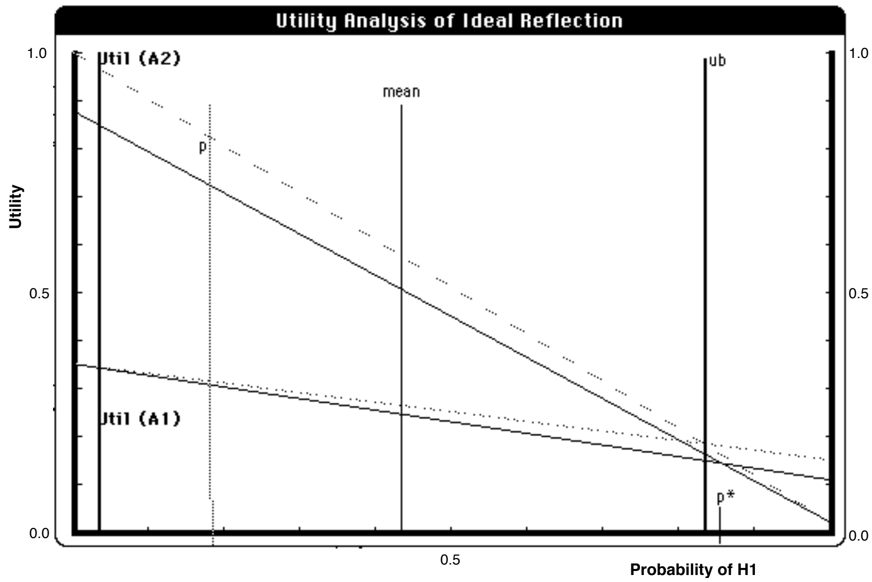
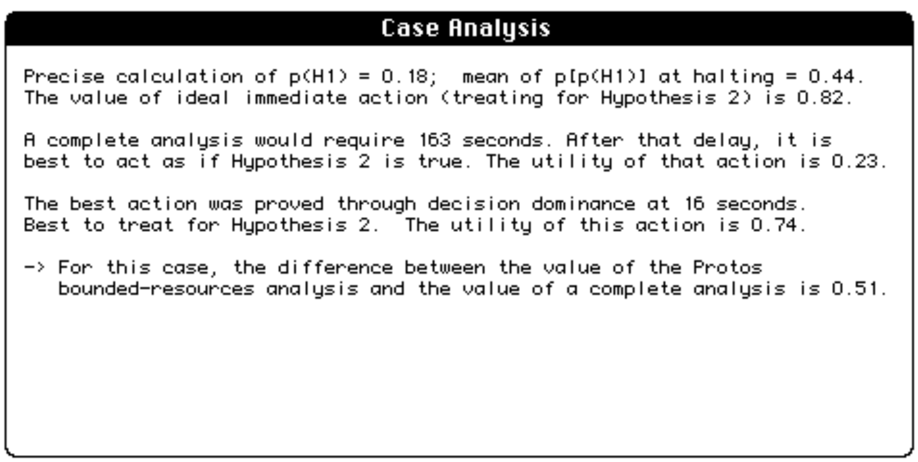


Figure 7.22: Utility analysis of decision and time for action recommended by Protos. This graph displays the utility of treating for pulmonary embolism [Util(A_1)] and for pneumonia [Util(A_2)] as a function of the probability of embolism (H_1). The best action, as indicated by the position of the upper bound on the probability of pulmonary embolism with respect to p^*, is to treat for pneumonia.



Case Analysis

Precise calculation of p(H1) = 0.18; mean of p[p(H1)] at halting = 0.44.
The value of ideal immediate action (treating for Hypothesis 2) is 0.82.

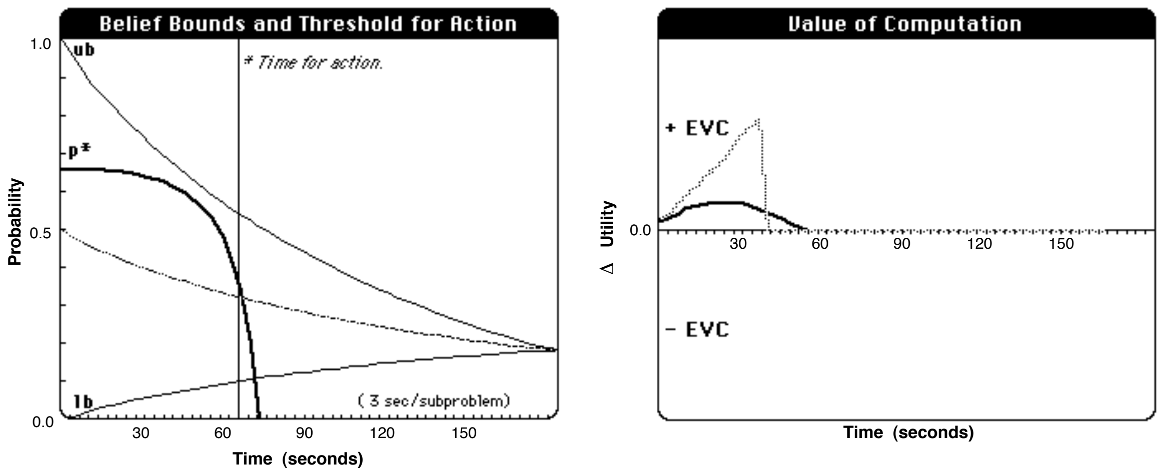
A complete analysis would require 163 seconds. After that delay, it is best to act as if Hypothesis 2 is true. The utility of that action is 0.23.

The best action was proved through decision dominance at 16 seconds.
Best to treat for Hypothesis 2. The utility of this action is 0.74.

-> For this case, the difference between the value of the Protos bounded-resources analysis and the value of a complete analysis is 0.51.

Figure 7.23: Case analysis.

The analysis summary indicates that Protos' threshold analysis is worth 0.51 more than a complete analysis, and only 0.08 less than instantaneous perfect computation.



(a) (b)

Figure 7.24: Revised time-dependent inference and ideal action.
The graph in (a) shows convergence of computed bounds on the probability of pulmonary embolism, and the time-dependent utility threshold. The graph in (b) displays the time-dependent (dark curve) and time-independent (lighter curve) EVC/BC for this evidential update. In this situation, Protos recommends action before a decision threshold is crossed.

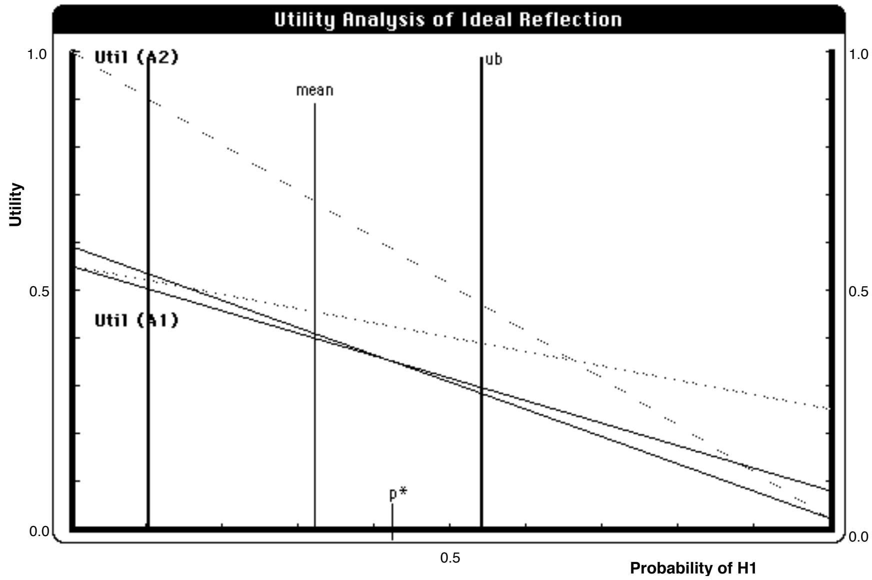
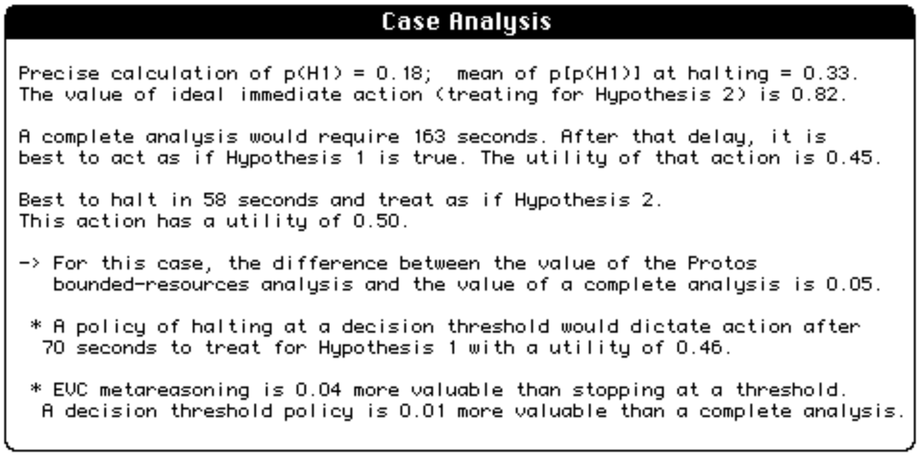


Figure 7.25: Utility analysis of decision and time for action recommended by Protos. This graph displays the utility of treating for pulmonary embolism [Util(A_1)] and for pneumonia [Util(A_2)] as a function of the probability of embolism (H_1). The position of the mean of the bounds interval, in relation to p^*, indicates that it is best to treat for pneumonia.

(lighter curve) for this evidential update. In this situation, Protos recommends that action should be taken before a decision threshold is crossed.

Figure 7.25 displays the utility of treating for pulmonary embolism [Util(A_1)] and for pneumonia [Util(A_2)] as a function of the probability of embolism (H_1), for the utility model in Table 7.8. The position of the mean of the bounds interval, in relation to p^*, indicates that the best action is to treat for pneumonia. The difference between the solid and broken utility lines indicate the time-dependent nature of the utility of the outcomes.

As indicated by the summary analysis in Figure 7.26, Protos' threshold analysis is worth 0.05 more than a complete analysis, and 0.04 more than the value of a threshold analysis. The dynamics of the decision threshold and the convergence on belief lead to different recommendations depending on the metareasoning strategy. Notice that an analysis based on the policy of waiting for a decision threshold to be reached would indicate that the ideal action is to treat for a pulmonary embolism. A policy of halting when the EVC becomes nonpositive indicates that the ideal action



Case Analysis

Precise calculation of p(H1) = 0.18; mean of p[p(H1)] at halting = 0.33.
The value of ideal immediate action (treating for Hypothesis 2) is 0.82.

A complete analysis would require 163 seconds. After that delay, it is
best to act as if Hypothesis 1 is true. The utility of that action is 0.45.

Best to halt in 58 seconds and treat as if Hypothesis 2.
This action has a utility of 0.50.

-> For this case, the difference between the value of the Protos
bounded-resources analysis and the value of a complete analysis is 0.05.

* A policy of halting at a decision threshold would dictate action after
70 seconds to treat for Hypothesis 1 with a utility of 0.46.

* EVC metareasoning is 0.04 more valuable than stopping at a threshold.
A decision threshold policy is 0.01 more valuable than a complete analysis.

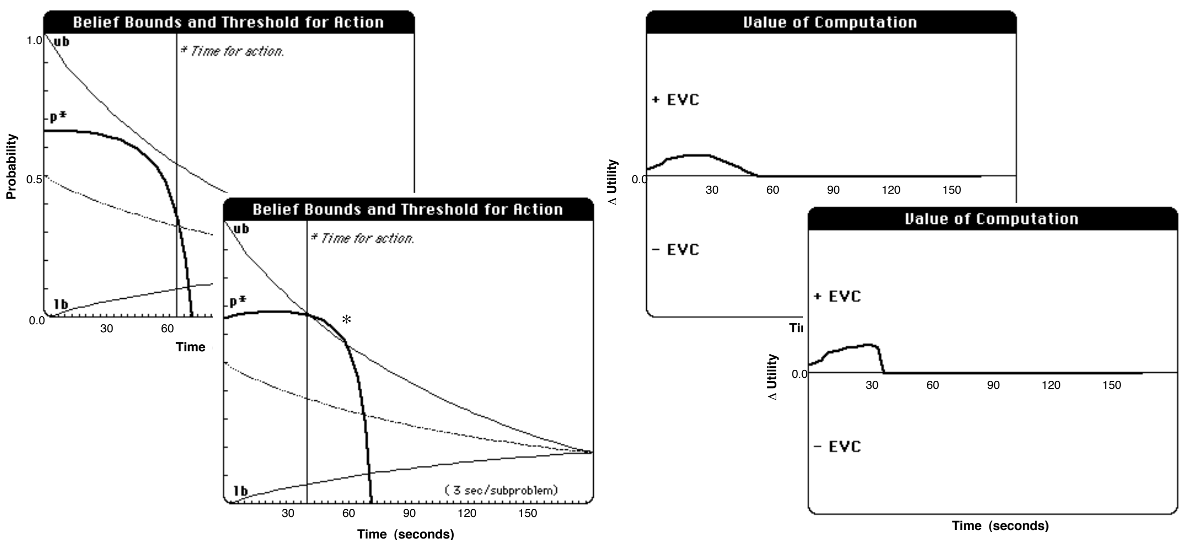
Figure 7.26: Case analysis.
The analysis summary indicates that Protos' analysis is worth 0.36 less than an immediate complete analysis. However, Protos' analysis is worth 0.05 more than a complete analysis, and is worth 0.04 more than the value of a threshold analysis.

is to treat for pneumonia. This result is in agreement with the recommendation of the immediate perfect analysis.

In a final example, we shall demonstrate again the sensitivity of ideal belief and action to subtleties in the structure of the utility model. Let us modify the utility model presented in Table 7.8 by increasing slightly the constant that specifies the cost of delay in treating for pulmonary embolism when this disease is present. The sequence of graphs in Figure 7.27(a) shows how changing the decay constant changes the bounded analysis, described in the previous example, to a threshold analysis. The graphs in 7.27(b) show the EVC/BC for the earlier and for the revised utility models. Figure 7.29 displays a closeup of the two probability refinement curves. The dynamics of the decision threshold and the convergence of belief dictate different Protos reasoning policies.

Figure 7.28 displays the change in the ideal halting point when the utility model presented in Table 7.8 is revised with information in the model displayed in the table at the top of Figure 7.27. The graph in 7.28(a) shows the initial bounded action, which is based on the location of the mean. In Figure 7.28(b), we see that a threshold has

Outcomes	U_{t_α}	Time dependence	Convergence
• Treat Pneumonia, Pneumonia	1.00	0.009	u(Embol,Pneu)
• Treat Embolism, Pneumonia	0.55	Static	0.55
• Treat Embolism, Embolism	0.25	0.020 → **0.025**	u(Pneu,Embol)
• Treat Pneumonia, Embolism	0.02	Static	0.02



(a) (b)

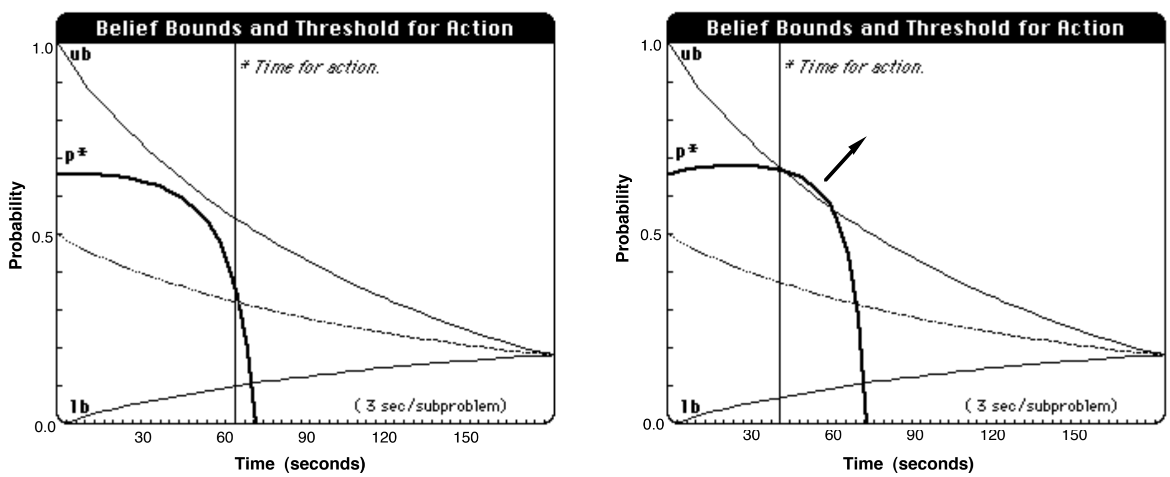
Figure 7.27: Sensitivity of reflection time to utility.
We demonstrate the sensitivity of ideal reflection and action to the details of the utility model and trajectory of inference. In this case, we change the utility model presented in Table 7.8 slightly, by increasing the decay constant that dictates the value of treating for a pulmonary embolism in the situation where an embolism is present. The graphs in (a) show how changing the decay constant changes Protos' previous bounded analysis (background) to a threshold analysis (foreground). The graphs in (b) show the EVC/BC for the earlier, and revised utility models.

Case Analysis

```
Precise calculation of p(H1) = 0.18;  mean of p[p(H1)] at halting = 0.38.
The value of ideal immediate action (treating for Hypothesis 2) is 0.82.

A complete analysis would require 163 seconds. After that delay, it is
best to act as if Hypothesis 1 is true. The utility of that action is 0.45.

The best action was proved through decision dominance at 37 seconds.
Best to treat for Hypothesis 2.  The utility of this action is 0.60.

-> For this case, the difference between the value of the Protos
   bounded-resources analysis and the value of a complete analysis is 0.16.
```

Figure 7.28: Case analysis.

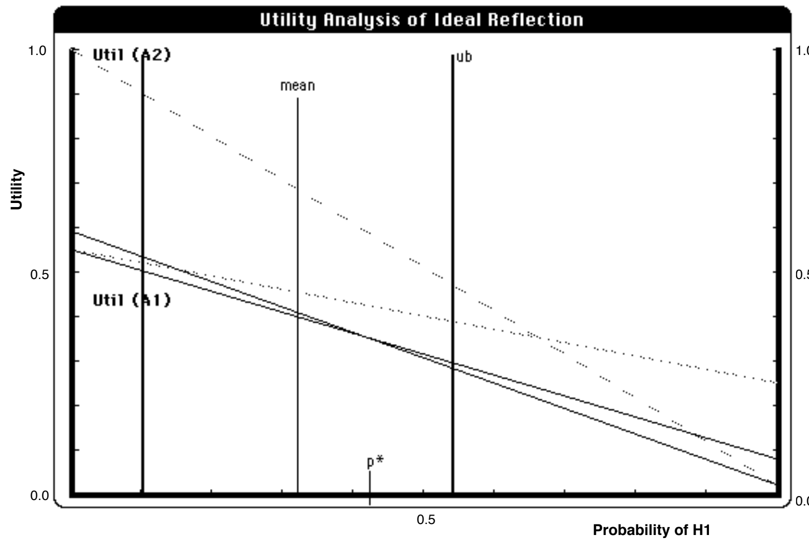
The analysis summary indicates that Protos' threshold analysis is worth 0.16 more than a complete analysis. Protos' recommended action and the action indicated by the ideal instantaneous analysis are in agreement: The best action is to treat for pneumonia. Note that, if the system were committed to computing a point probability, the optimal choice would be to treat for a pulmonary embolism.



(a) (b)

Figure 7.29: A closeup of graphs displaying sensitivity of reflection time to utility. By increasing the decay constant that dictates the time-dependent utility of treating a patient for a pulmonary embolism in the situation where an embolism is present, we change Protos' reasoning from (a) a bounded analysis to (b) a threshold analysis.

(a)



(b)

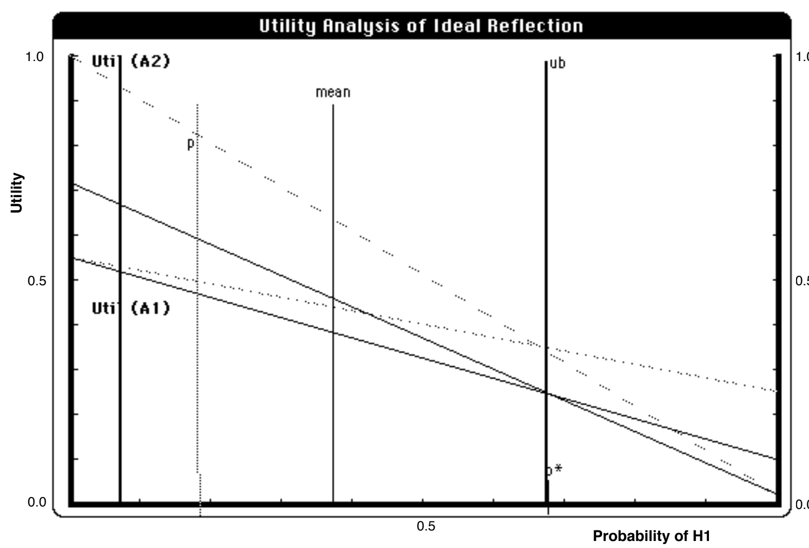


Figure 7.30: Comparative utility analyses of ideal reflection.

This graph displays the change in the ideal halting point with revising the utility model. The graph in (a) shows the initial bounded action, based on the mean. In (b), we see that a threshold has been reached.

been reached.

The analysis summary indicates that Protos' threshold analysis is worth 0.16 more than a complete analysis. The Protos analysis and the ideal analysis are in agreement: It is best to treat for pneumonia. Note that, by the time exact inference would be completed, the optimal choice would be to treat for a pulmonary embolism (H_2). Given the sensitivity of the utilities of alternate outcomes to delay, the best action to take can depend on the time that action occurs.

The phenomenon of an optimal decision changing with delay has been identified previously in the context of medical decision analysis (McNutt and Pauker, 1987). In this work, investigators demonstrated how the optimal medical therapy indicated by a decision analysis could change radically, given the delay required to perform that analysis. The study emphasized how an ideal decision could be sensitive to time-dependent changes in the values of important probabilities in a decision model (e.g., the probability of a rebleed occuring at different times after an initial hemmorhage). The values of these probabilities do not change because of computational activity, as in Protos' application of bounded conditioning; rather, the probabilities change because of causal processes. Modeling such time-dependent probabilities in belief networks can introduce greater richness to the dynamics of Protos' beliefs and actions under bounded resources.

7.3 Summary

In this chapter, I defined the value of metareasoning (EVM) and demonstrated the behavior of Protos in time-dependent decision contexts. I presented illustrative examples of time-pressured medical decision making with several belief networks. To compute the value of Protos' recommended actions, I made use of reference utilities computed from gold-standard probabilities. Point probabilities, obtained by solving inference problems completely, served as the gold-standard beliefs. For the cases investigated, Protos' case summarizer computed the value of Protos' actions in relation to a complete analysis with bounded conditioning, and in relation to a decision-threshold policy. For the decision-threshold policy, we determined the expected utilities of the best actions to take when an upper or lower bound on a probability reached a decision threshold. The EVM analyses demonstrated that the Protos metareasoning techniques can provide valuable control of inference for approximation strategies, and showed that the behavior of Protos is sensitive to subtleties in the evidential updating and time-dependent utility model associated with a case. Protos' procedures for monitoring the value of continuing to deliberate versus that of taking more timely action appear to be especially useful when the expected utility of the best decision diminishes significantly with delay.

Chapter 8

Cognitive Resources as Constraints in Normative Reasoning

A criticism of the simple probabilistic diagnostic systems, developed by medical-informatics investigators in the 1960s, is that clinicians find their reasoning strategies unnatural and their recommendations difficult to understand (Szolovits, 1982; Politser, 1984). A fundamental source of difficulty in the comprehension and explanation of decision-theoretic inference is the inescapable complexity of many computer-based normative analyses. I have viewed problems with the understandability of normative reasoning as arising from constraints on the cognitive resources of people. As a complement to my investigation of the value of normative reasoning under computational resource constraints, I have studied techniques for flexibly trading off the opacity of detailed analyses for simpler and clearer, but less precise, computation. As displayed in Figure 8.1, this work can be viewed as being analogous to the investigation of flexible normative reasoning under time constraints. Here, we

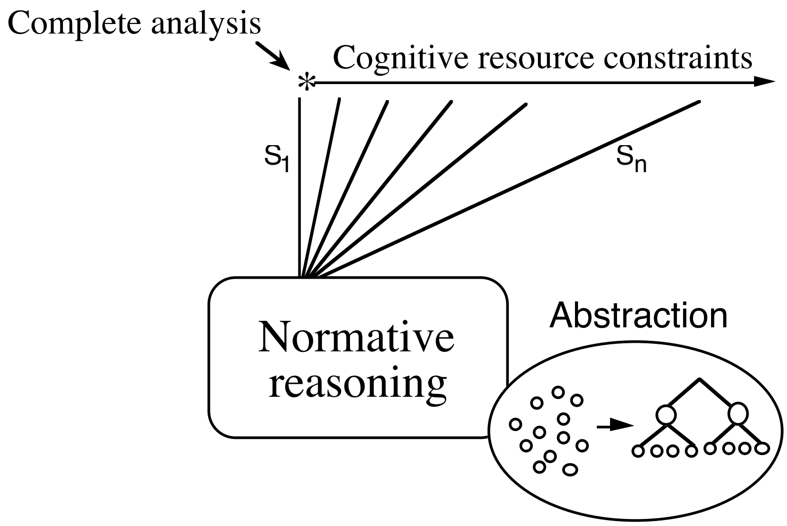


Figure 8.1: Normative reasoning under cognitive-resource constraints.
We consider the effect of cognitive-resource limitations on the ability of a person to make optimal use of a normative analysis. By developing flexible methods that trade off the optimality of a normative analysis with the simplicity and, thus, the understandability of that analysis, we can produce computer-based reasoning strategies that have greater value to people.

seek to increase the value of normative analyses by considering constraints on cognitive resources that may limit the ability of people to understand—or accept—the conclusions of normative reasoning.

In this chapter, I shall describe research on the imposition of bounds on the complexity of normative reasoning with a goal of making reasoning more natural and easy to explain. I shall describe a facility that allows a user to abstract sets of distinctions into smaller numbers of propositions at higher levels of abstraction. The abstraction facility has been integrated into Pathfinder, a reasoning system that aids pathologists in making surgical-pathology diagnoses. The approach enables a user to impose human-oriented grouping strategies on complex value-of-information analyses. A user can modulate the detail and perspective of an analysis to reflect different cognitive styles. This capability can increase the naturalness of normative inference and explanation, without incurring significant losses in the optimality of recommendations.

8.1 Managing the Complexity of Normativity

The clarity of a computer-based reasoning system can affect the value of a system to a user. Surveys of the preferences of clinicians have identified clarity of reasoning as an important factor in the acceptance of a reasoning system (Teach and Shortliffe, 1981; Buchanan, 1982; Buchanan and Shortliffe, 1984). The importance of reasoning transparency in expert systems has made explanation an central concern of many AI investigators (Shortliffe, 1982). The inability of the probabilistic reasoning systems to handle inference and decision problems in a natural manner has been attributed to the inadequate expressiveness and inflexibility of the systems (Gorry, 1973; Davis, 1982).

Most investigators have attempted to explain the recommendations of normative reasoning by developing facilities for summarizing salient features of an analysis. For example, Langlotz (Langlotz et al., 1986) and Klein (Klein, 1987) have built systems that construct qualitative summaries of dominant tradeoffs and key results of sensitivity analyses. I have studied the simplification of normative analyses through the abstraction of the diagnostic entities considered by a system. In particular, I have worked to develop tools that can allow a normative system to control dynamically the complexity of a normative analysis by abstracting a set of detailed distinctions into *groups* of entities. These groups can be viewed as distinctions at a higher lever of abstraction. For example, we can group all non-malignant diseases, considered by a medical expert system, into a category named *benign diseases* and can manipulate this group as a single distinction in a less-detailed, approximate normative analysis.

Increasing the comprehensibility (and thus, the value) of normative analyses through simplification was suggested initially by observing the diagnostic behavior of several pathologists involved with the Pathfinder project. Our work has been further supported by findings of cognitive psychologists. Psychology experiments have demonstrated repeatably that people have severe limitations in their ability to consider more than a handful of concepts in the short term (Bruner et al., 1956). One study demonstrated that people cannot effectively comprehend more than 7 ± 2 concepts in the short term (Miller, 1956). Another study found that humans cannot retain and reason about more than *two* concepts in an environment with distractions (Waugh and Norman, 1965).

Other cognitive psychology studies have demonstrated that people tend to use groups and hierarchies of groups to reduce the number of entities under consideration (Simon, 1973b). Psychologists have speculated that hierarchies of abstraction are used often by people to facilitate easy indexing through relationships among classes at different levels of abstraction (Mesarovic et al., 1970). Investigators have identified the use of hierarchies of groups of distinctions in a variety of domains. Of particular relevance to our work, psychologists studying medical decision making have found that physicians in specialty areas of medicine frequently make use of abstraction strategies for managing the complexity of clinical problem solving (Elstein et al., 1971; Elstein et al., 1978).

In AI, investigators have studied the use of grouping in the control of reasoning in rule-based systems (Clancey, 1985). There has also been independent discussion of the usefulness of grouping in probabilistic reasoning systems (Ben-Bassat and Teeni, 1984). My work on the abstraction of normative analyses has focused on the introduction of flexibility to the value-of-information machinery in the Pathfinder reasoning system. I shall first describe the calculation of the value of computation for detailed and abstract models. Then, I shall introduce the Pathfinder project and domain, and describe an abstraction facility developed for the Pathfinder system.

8.2 Value of Information

Let us consider how we can might compute the value of information for selecting observations or tests, and for deciding when it is best to cease evidence gathering, and to take an action in the world. We use $u(A_i, H_j)$ to represent the utility of a decision maker who takes an action (or set of actions) A_i when state H_j is true. Recalling Equation 4.1 introduced in Chapter 4, given a probability distribution $p(H|E_k, \xi)$ over a set of mutually exclusive and exhaustive hypotheses, conditioned on evidence

E_k, and on background state of information ξ, the expected utility of an action A_i is

$$\mathrm{eu}(A_i, p(H|E_k, \xi)) \;=\; \sum_{j=1}^{n} p(H_j|E_k, \xi) u(A_i, H_j)$$

The ideal decision, A^*, is the action with the greatest expected utility, given the probability distribution and the utility model,

$$A^* \;=\; \arg\max_{A_i} \sum_{j=1}^{n} p(H_j|E_k, \xi) u(A_i, H_j) \tag{8.1}$$

Let us assume that an agent has already made a set of observations, $\mathbf{E}$. The agent can perform additional tests, E_k, each associated with a set of mutually exclusive and exhaustive values or outcomes (e.g., true, false). We shall use E_k^l to represent a test result or sensor value, where l indexes alternate outcomes of the test or observation. We denote the values of E_k, by $E_k^1, \ldots, E_k^m$, where m is the number of mutually exclusive values.

We can compute the expected value of information (EVI) of performing a test or making an observation by conditioning the probability of different states of the world on different outcomes of the test, and determining the expected value of the best actions associated with the revised probability distributions $p(H|E_k^l, \mathbf{E}, \xi)$. We weight the expected utility, associated with each test outcome, by the probability of that outcome, $p(E_k^l|\mathbf{E}, \xi)$. Then, we consider the difference between the expected utility of actions, dictated by the current state of information, and the expected utility of acting after performing a test. Finally, we see if the informational value of a test exceeds the cost of the test, $\mathcal{C}(E_k)$. For expected value decision makers, the value of information is

$$\begin{aligned} \mathrm{EVI}(E_k) \;=\; & \sum_{l=1}^{m} p(E_k^l|\mathbf{E}, \xi) \left[\max_{A_i} \sum_{j=1}^{n} p(H_j|E_k^l, \mathbf{E}, \xi) u(A_i, H_j) \right] \\ & - \max_{A_i} \sum_{j=1}^{n} p(H_j|\mathbf{E}, \xi) u(A_i, H_j) \\ & - \mathcal{C}(E_k) \end{aligned} \tag{8.2}$$

We can generalize Equation 8.2 to consider the time associated with gathering information in critical situations by explicitly considering the time-dependent nature of utility. We substitute time-dependent utility $u(A_i, H_j, \mathcal{T}(E_k))$ for time-dependent utility, $u(A_i, H_j)$, where $\mathcal{T}(E_k)$ is the time required to perform test E_k

Note that we formulated the value-of-information in terms of evaluating the value of a single piece of evidence. More generally, we should consider the value of all subsets of future observation and consider the value of alternate sequences of tests, and about the value of stopping versus continuing to gather evidence. Unfortunately, such a general analysis is intractable. Thus, myopic analyses of the *next best* test to perform are employed frequently in practice. Myopic calculations are based on the assumption that an action will be taken after making one additional observation. This assumption is often violated by decisions to perform additional myopic analyses. Myopic analyses of the value of information can lead to erroneous decisions to halt evidence gathering: Situations can arise where a myopic analysis cannot identify a single test with positive value of information, yet where a combination of observations may have positive value. Although we have a poor understanding of the problems associated with myopic analyses in practice, there is evidence that myopic analyses can offer a good approximation in some domains. Gorry and Barnett explored the value of considering larger sequences of tests for a program built to assist physicians with diagnosing congenital heart disease (Gorry and Barnett, 1968). They demonstrated that the myopic analysis does not significantly affect the diagnostic accuracy of the program.

8.3 Varying the Level of Abstraction

Given constraints on cognition or computation time, a detailed model may be more difficult to understand or to solve, and, thus, less valuable than a smaller, more abstract model. We shall focus on the simplification of value-of-information analyses through abstraction. We can modulate the level of detail of a value-of-information analysis by substituting atomic actions A_i or states of the world H_j with more abstract distinctions.

A common form of abstraction is the clustering of states of the world into *sets* of states that share common properties. Some sets may form natural, mutually exclusive groups, such as *infectious* and *inflammatory* diseases in medicine. Given a set of mutually exclusive states, we can reason about groups in terms of the truth of disjunctions of their elements. For example, in medical diagnosis, we might abstract six mutually exclusive diseases $H_1, \ldots, H_6$ into two disease groups, G_1 and G_2, and reason about belief in the disjunctions $H_1 \vee H_3 \vee H_5$ and $H_2 \vee H_4 \vee H_6$ instead of all six diseases. The probability assigned to groups of mutually exclusive states is the sum of the probabilities of the members of each group. We can compute the value of information for discriminating among a set of such groups by substituting distinctions H_j with groups G_j in Equation 8.2. We need to consider a group utility model with preference information, $u(A_i, G_j)$, and probability distributions over groups, $p(G_j|E_k^l, \mathbf{E}, \xi)$ and $p(G_j|\mathbf{E}, \xi)$, where

$$p(G_j|E_k^l, \mathbf{E}, \xi) \;=\; \sum_{x \in G_j} p(H_x|E_k^l, \mathbf{E}, \xi)$$

We can build hierarchies of abstraction by viewing groups as elements in more abstract sets. For example, the infectious and inflammatory groups might be elements of the set *benign*, and we may wish to select tests that can distinguish benign diseases from malignant diseases. The recommendations generated by computing the value of information at higher levels of abstraction can be more understandable and explainable. We can modulate the complexity of value-of-information analyses in automated reasoning systems by developing utility models for sets of distinctions at different levels of detail. Let us explore the use of alternative utility models and levels of abstraction for value-of-information reasoning in the Pathfinder diagnostic system.

8.4 The Pathfinder Project

The Pathfinder project was initiated to solve problems that general pathologists have making hematopathology diagnoses. Pathfinder investigators have focused on the development of representations that allow for the efficient acquisition of probabilistic knowledge from experts (Heckerman et al., 1985; Heckerman et al., 1989b; Heckerman et al., 1990; Heckerman, 1990b). However, we also explored the control of value-of-information analyses through abstraction.

A goal of the Pathfinder project was the construction of a computer-based system to guide pathologists in the interpretation of histologic features that appear in sections of lymph-node tissue. The microscopic interpretation of lymph-node biopsies is a difficult task for surgical pathologists (Kim et al., 1982; Velez-Garcia et al., 1983). The 30 malignant diseases of the lymph nodes have to be distinguished from approximately 30 benign diseases, many of which closely resemble malignant lymphomas. The accurate diagnosis of diseases that present as complex visual patterns in lymph-node tissue is crucial for the determination of prognosis and therapy; most malignant lymphomas have a distinctive natural history, response to therapy, and survival rates (Rosenberg, 1985).

The computational architecture of the Pathfinder system is based on the *hypothetico-deductive* approach to diagnosis (also called the *method of sequential diagnosis* in the medical-informatics literature (Gorry and Barnett, 1968; Gorry, 1973)). A flowchart representation of this method is shown in Figure 8.2. Hypothetico-deductive systems allow a user to input a small set of salient manifestations. The systems assign belief to competing hypotheses, and then recommend the best next observations to make or tests to perform. New information that becomes available is considered in generating a revised belief assignment. The belief-assignment—information-gathering cycle continues until a decision is made to halt additional information gathering. In Pathfinder, belief is assigned by performing probabilistic inference on a belief network that represents dependencies among histologic features and diseases. The system employs value-of-information analyses to select the next best questions to ask, and to decide when a diagnostic decision should be made.

The current version of Pathfinder considers approximately 60 diseases of lymph nodes, constructing differential diagnoses through the consideration of evidence about the status of up to 120 microscopic *features* presenting in lymph-node tissue. A pathologist enters into the Pathfinder system information about the status of one or more features seen on a tissue biopsy. The histologic features are each structured into a set

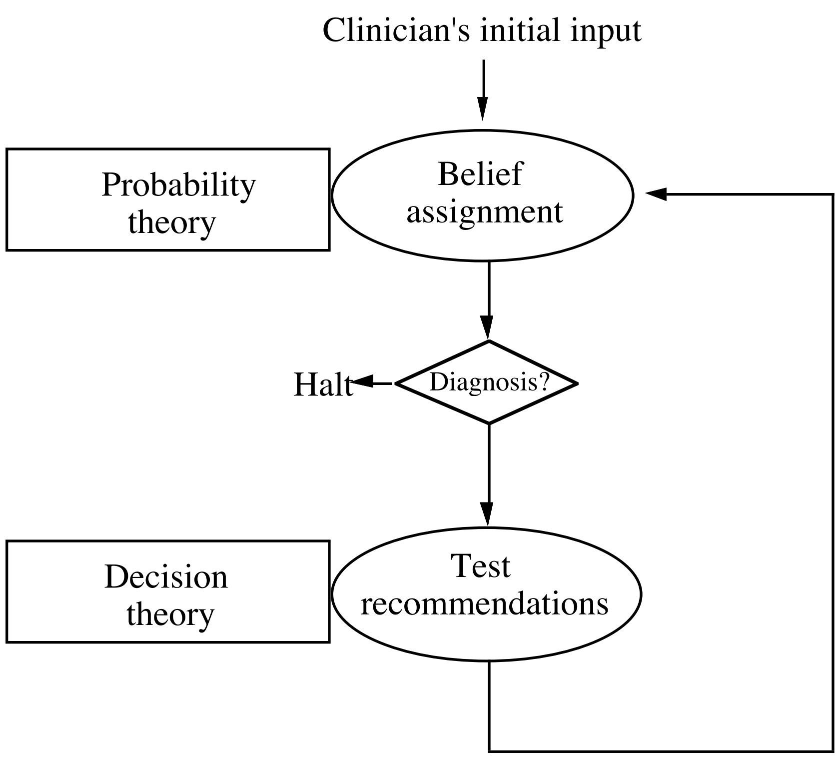


Figure 8.2: Hypothetico-deductive reasoning.
Pathfinder and other hypothetico-deductive systems assign belief to competing hypotheses, based on a small set of initial observations. Then, they recommend the best next observations to make. The systems continue to cycle between belief assignment and test recommendation until a decision is made to make a diagnostic decision.

of mutually exclusive and exhaustive values. For example, the feature PSEUDOFOLLICULARITY can take on any one of the values *Absent*, *Slight*, *Moderate*, or *Prominent*. After the initial features are entered, a probabilistic reasoner assigns probability to alternative diseases considered by the system. At the user's request, the system then applies a value-of-information analysis to generate new test recommendations. The process continues until continuing to gather evidence is no longer valuable.

8.4.1 A Base-Level Utility Model for Diagnosis

Diagnostic reasoning systems, such as Pathfinder, are built to assist people with inference about the state of a system or patient, rather than to consider explictly alternative therapies. For considering abstraction in diagnosis, we shall use $u(H_i, H_j)$ to refer to the utility of taking actions appropriate for treating disease H_i when disease H_j is present. The $u(H_i, H_j)$ in the utility model of a diagnostic system should reflect a patient's preferences. Pathologists typically do not meet with patients and, thus, must estimate the preferences of their patients. For the Pathfinder utility model, we asked an expert pathologist to imagine that he was the patient, and to provide the $u(H_i, H_j)$ (Heckerman, 1990a). We employed a measure of utility developed by Howard (Howard, 1980), described in Chapter 4. We refer to this base-level model of patient preferences as Pathfinder's *healthcare* utility model. Pathfinder uses these measures of utility in a myopic value-of-information analysis to order its recommendations for gathering additional information about the microscopic appearance of a tissue section.

8.4.2 A Finer-Grained Utility Model for Discrimination

Some pathologists work to increase our understanding of the clinical significance of subtle disease distinctions. They are interested in identifying as many subtypes of disease as possible, in the hope that medical researchers will develop more specific therapies. Thus, they are interested in discriminating diseases, even if the diseases are believed currently to have the same therapy and prognostic course. For this case, a utility model for directing the collection of discriminatory evidence in a value-of-information analysis should treat all distinctions as being equally important. Such a fine-grained utility model directs evidence gathering so as to distinguish among all disease entities. In a fine-grained model *discrimination model*, we consider $u(H_i, H_i) = 1$ and $u(H_i, H_j) = 0$ for $H_i \neq H_j$. This fine-grained utility model is useful for clinical trials and for training.

Pathfinder makes a fine-grained discrimination model available to users. The system employs an efficient information-theoretic approximation to the computation of value of information for the fine-grained model that makes use of a measure of information called *entropy* (Heckerman et al., 1985; Horvitz et al., 1989b; Heckerman et al., 1990). Given assumptions of equal cost of tests, and equal cost of making an error, Ben-Bassat has shown that tests selected by a general value-of-information computation are nearly identical to those generated by entropy calculations (Ben-Bassat, 1978). Other researchers have used this approximation in medical expert systems (Gorry, 1973).

8.4.3 General Abstraction of Diagnostic Hypotheses

In developing Pathfinder, we found that pathologists often work at levels of abstraction that include, as specific cases, the most detailed level of analysis, provided by the fine-grained discrimination model, and the healthcare model (Heckerman et al., 1985; Horvitz et al., 1986b). Pathologists reason about groups of related diseases for managing the complexity of diagnostic inference and for guiding the acquisition of information.

As demonstrated in Figure 8.3, we found that pathologists can generate abstractions dynamically by grouping similar diseases. Measures of similarity range from shared morphologic features to shared membership in broad classes of disease, such as *infectious* and *inflammatory*. Beyond the creation of groups based on similarity, we found that pathologists my simplify the discrimination task by employing informational grouping strategies. For example, at a particular point in a diagnostic session, a pathologist may wish to gather information that discriminates between the disease with the leading probability, and a group containing all other diseases. Alternatively, he may wish to distinguish the top two malignant diseases from all others.

We found that pathologists tend also to make use of hierarchies of abstraction by

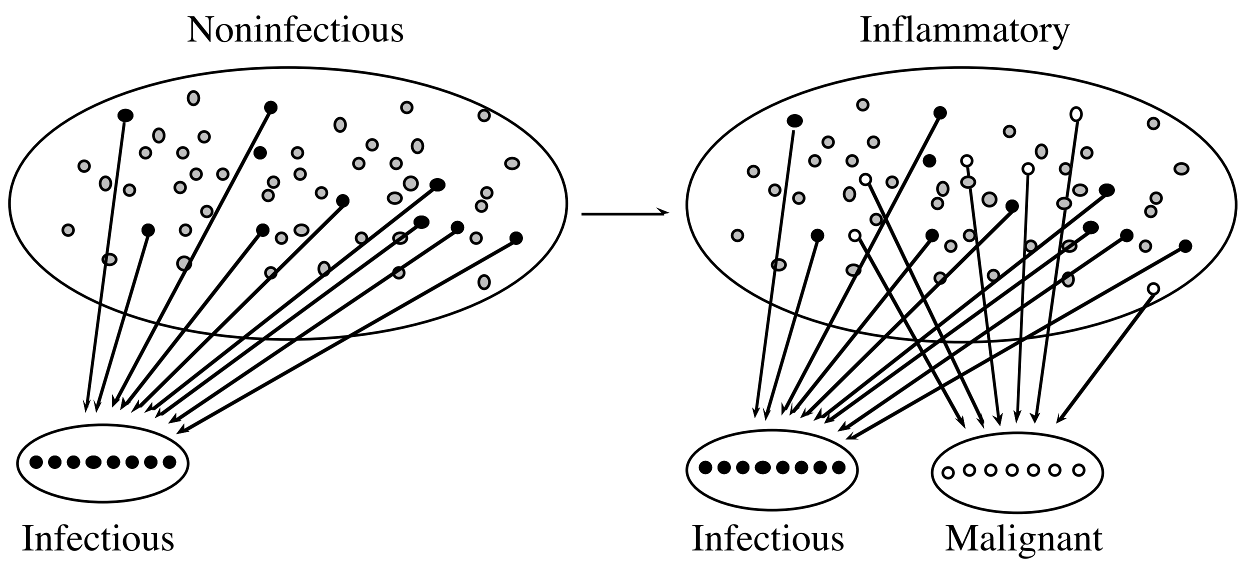


Figure 8.3: Creation of useful distinctions with grouping.
We have assessed from users useful groups of diseases that can simplify value-of-information reasoning and explanation. Groups can be defined by etiological or morphological similarity, or can be based on more abstract notions, such as distinguishing the disease with the highest probability from other hypotheses.

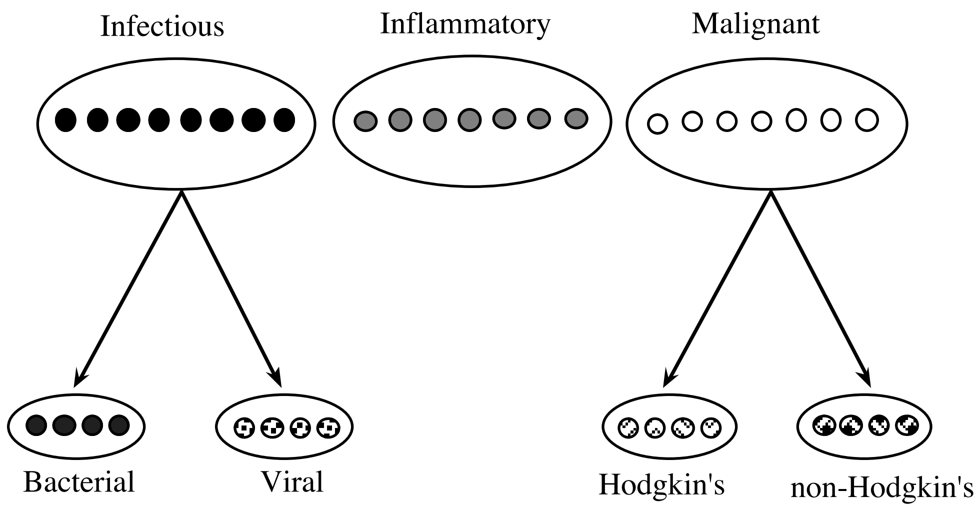


Figure 8.4: Representing hierarchies of abstraction.
Hierarchies of groups of disease hypotheses can capture knowledge about the order of analysis preferred by a decision maker. We can analyze a relatively constant number of hypotheses throughout a diagnostic case.

nesting groups, as portrayed in Figure 8.4 Pathologists' evidence-gathering strategies often could be described by the traversal of hierarchies. Hierarchies of disease groups allow pathologists to keep small the number of entities being considered at any time, while guiding diagnosis to increasingly detailed levels. One such disease hierarchy is shown in Figure 8.5. When using this hierarchy, a pathologist first considers features that discriminate between only benign and malignant diseases. If the differential diagnosis is narrowed to only malignant diseases, the pathologist then discriminates between primary and metastatic diseases. If metastatic diseases are ruled out, the pathologist considers features that discriminate between nonHodgkin's and Hodgkin's lymphomas. Finally, when all diseases under consideration are in the same group, a pathologist discriminates among these diseases individually.

We found that pathologists can use many different abstraction hierarchies, each representing alternative *perspectives* on any given diagnostic problem; that is, there are multiple ways to manage the complexity of diagnostic reasoning through abstraction (Horvitz, 1987a; Horvitz et al., 1989b). Figure 8.6 shows a strategy employed

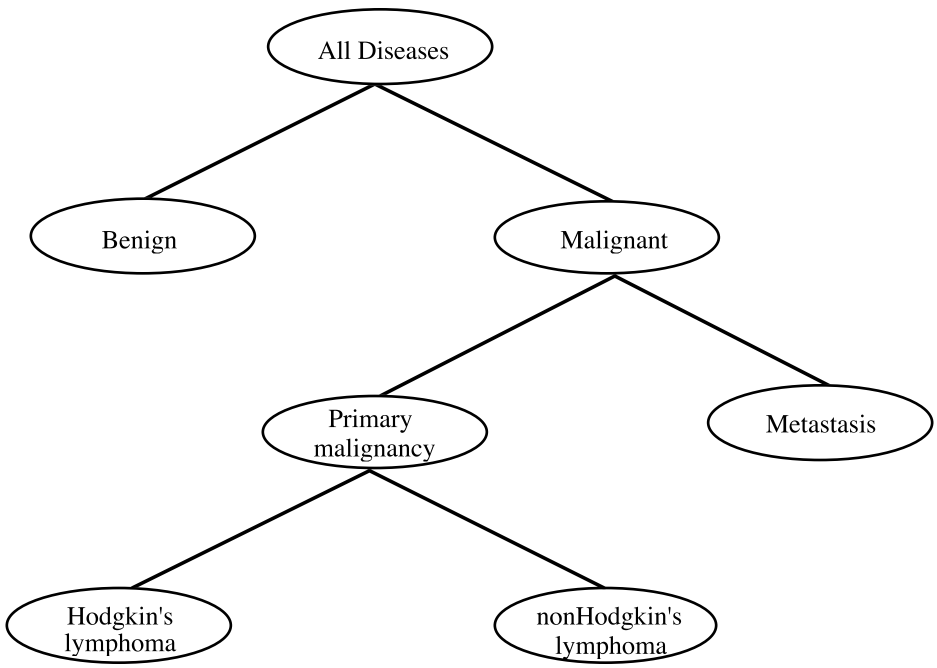


Figure 8.5: Classifying diseases into an etiological hierarchy.
This heuristic problem-solving hierarchy portrays how a pathologist may categorize diseases into a sequence of abstraction classes to manage the complexity of diagnostic inference.

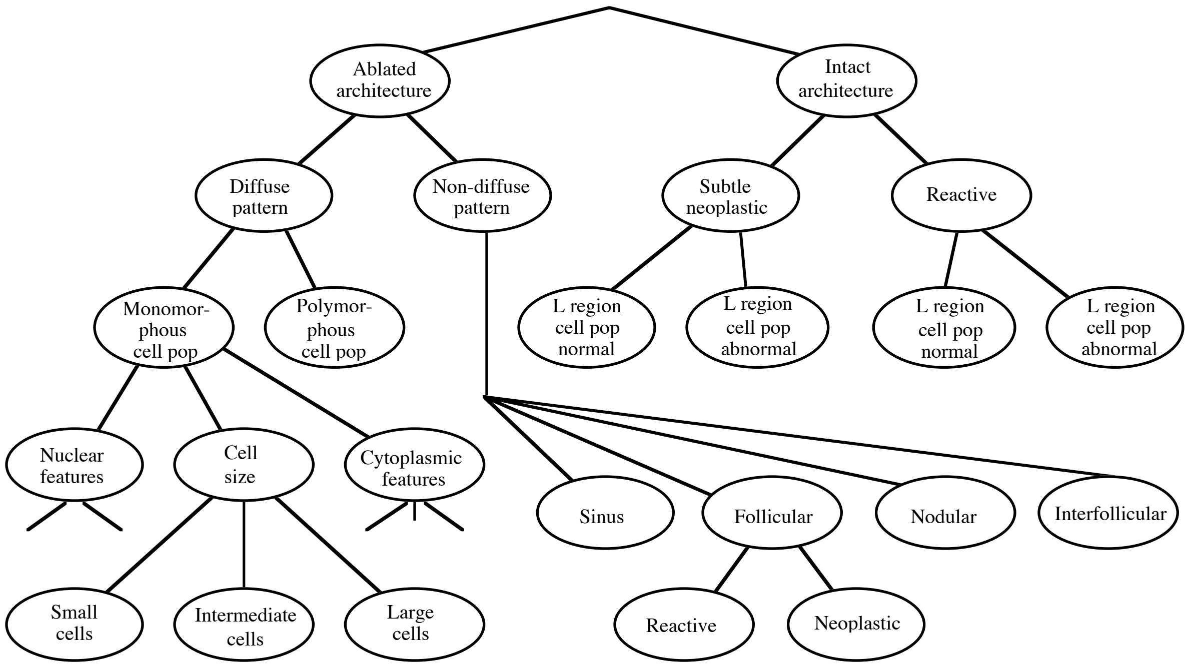


Figure 8.6: A different perspective on diagnosis.
This abstraction hierarchy represents the formulation of the diagnostic problem from the perspective of the morphological *pattern* seen in a lymph-node section (*pop* is an abbreviation for population).

by a hematopathology resident who was singled out by our domain expert as having a special gift for the diagnosis of lymph-node pathology. This grouping strategy stresses the use of high-level morphological *patterns*. Figure 8.7 shows a classification strategy based on the *origin* of the dominating population of proliferating cells. The chief expert on the Pathfinder project used this strategy.

8.5 An Abstraction Facility for Pathfinder

As highlighted in Figure 8.8, we worked to introduce flexibility into Pathfinder by developing tools that enable the system to discriminate among hypotheses created by alternative groupings of diseases. The use of abstraction hierarchies by pathologists participating in Pathfinder research can be viewed as alternative coarsenings of the fine-grained utility model. Rather than apply the value-of-information calculations

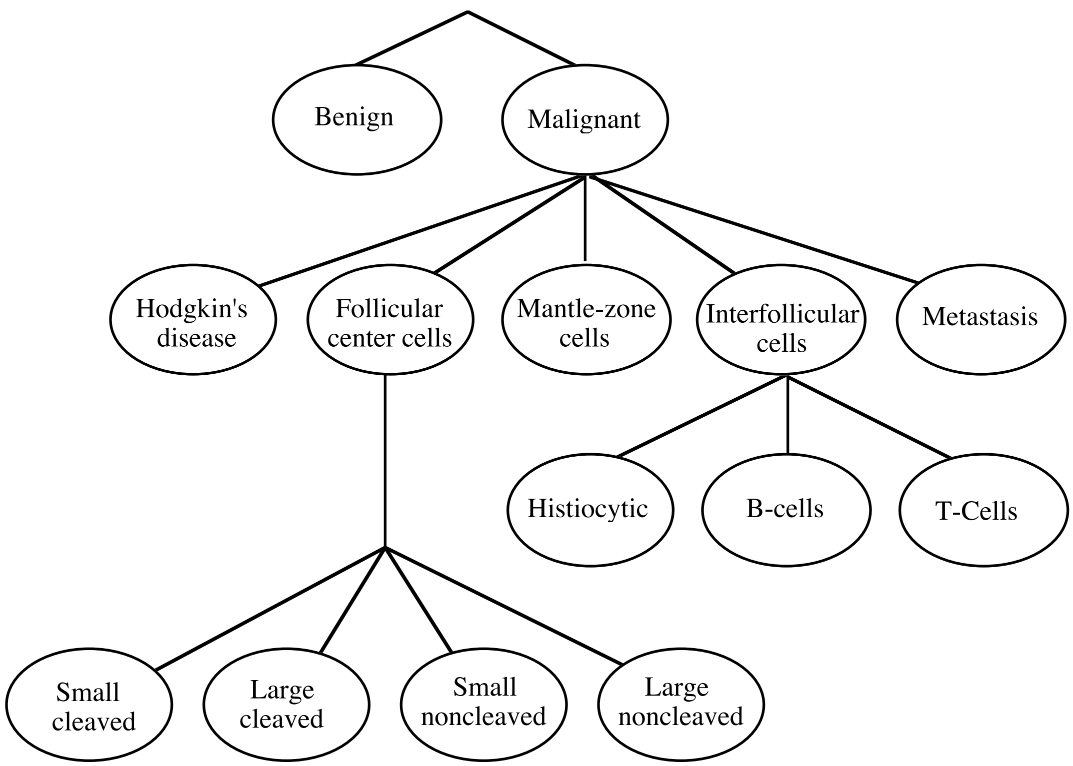


Figure 8.7: Abstraction based on predominating cell line.
This abstraction hierarchy represents the formulation of the diagnostic problem from the perspective of the *origin* of the predominant proliferating cell line.

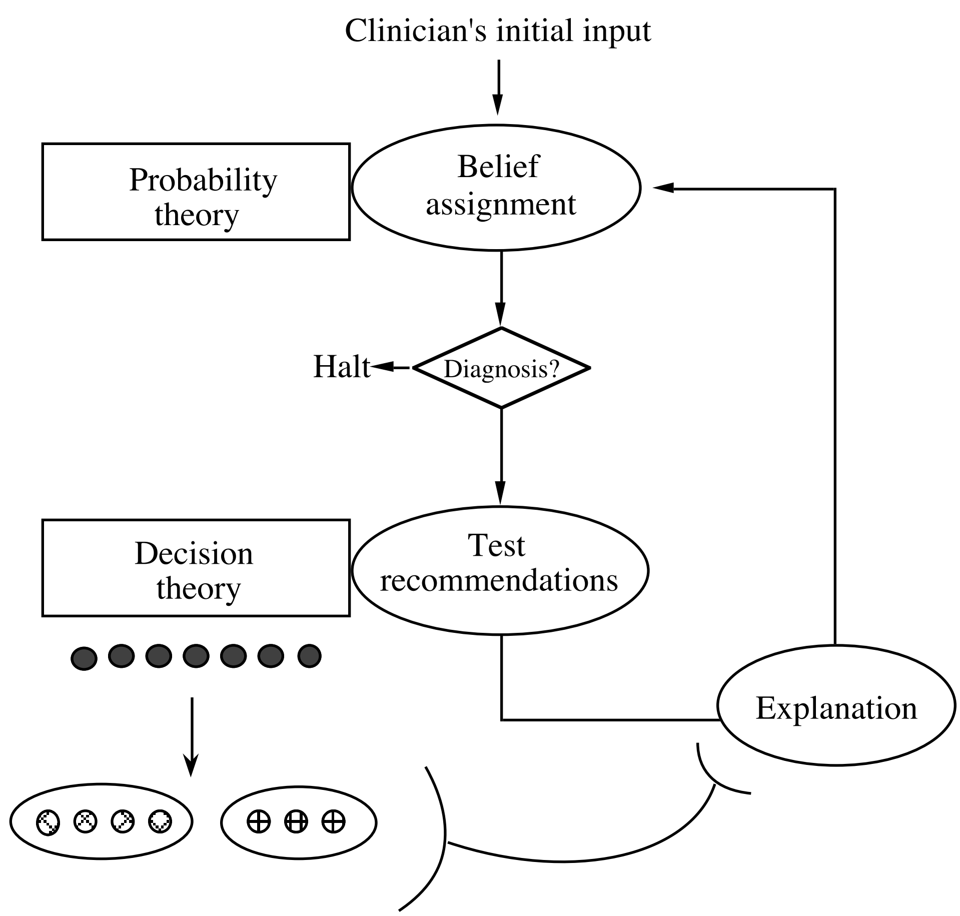


Figure 8.8: Addition of heuristic abstraction to Pathfinder's reasoning. Instead of forcing reasoning at the most detailed model, the abstraction facility allows for the arbitrary grouping of distinctions in the value-of-information calculation.

to single diagnostic entities, we apply the calculations to groups, considering the probability of each group, and the utility of misdiagnoses among groups of diseases.

We can construct different forms of group utility model. We can employ a *group-discrimination* model, analogous to the fine-grained discrmination model. For this model, we consider $u(G_i, G_i) = 1$ and $u(G_i, G_j) = 0$ for $G_i \neq G_j$. Creating this utility model does not require any additional assessment. For a grouped analysis that is patient-oriented, we may wish to assess ahead of time sets of utilities for several different abstraction hierarchies. However, we may wish to allow a physician, during a case analysis, to dynamically define groups of hypotheses that reflect his current perspective on the problem. For these situations, it typically is not feasible to assess the required utilities ahead of time. We can approximate utilities for group analyses dynamically from the base-level utilities $u(H_i, H_j)$ for single diseases. For example, we can consider the group utility $u(G_1, G_2)$ to be the average of utilities $u(H_i, H_j)$, where $H_i \in G_1$ and $H_j \in G_2$. Alternatively, we can take the utility to be the minimum of base-level utilities $u(H_i, H_j)$, such that H_i and H_j are elements of G_1 and G_2, respectively.

I developed a general abstraction facility and worked with colleagues on the Pathfinder project to integrate that facility into the Pathfinder expert system (Horvitz et al., 1989b). The facility allows system developers or users to specify a library of intuitive classes of diseases and to nest these abstractions within arbitrary strategic scripts. Figure 8.9 displays a single-level abstraction in Pathfinder. In this case, an ungrouped Pathfinder differential diagnosis is divided into groups of benign diseases, lymphomas, and metastatic diseases. The system lists the diseases of each group by likelihood and displays, for each group, the probability that the true disease is in contained that abstractions.

Pathfinder allows users to examine a diagnostic problem simultaneously from different perspectives. Multiple windows—each representing a different perspective on the same problem—can be invoked. The windows displays the differential diagnosis and highlight the current level of abstraction. By clicking on one of the windows, a user activates the perspective. The evidence-gathering strategies, based on alternative group utility models introduces human-oriented flexibility to the generation and

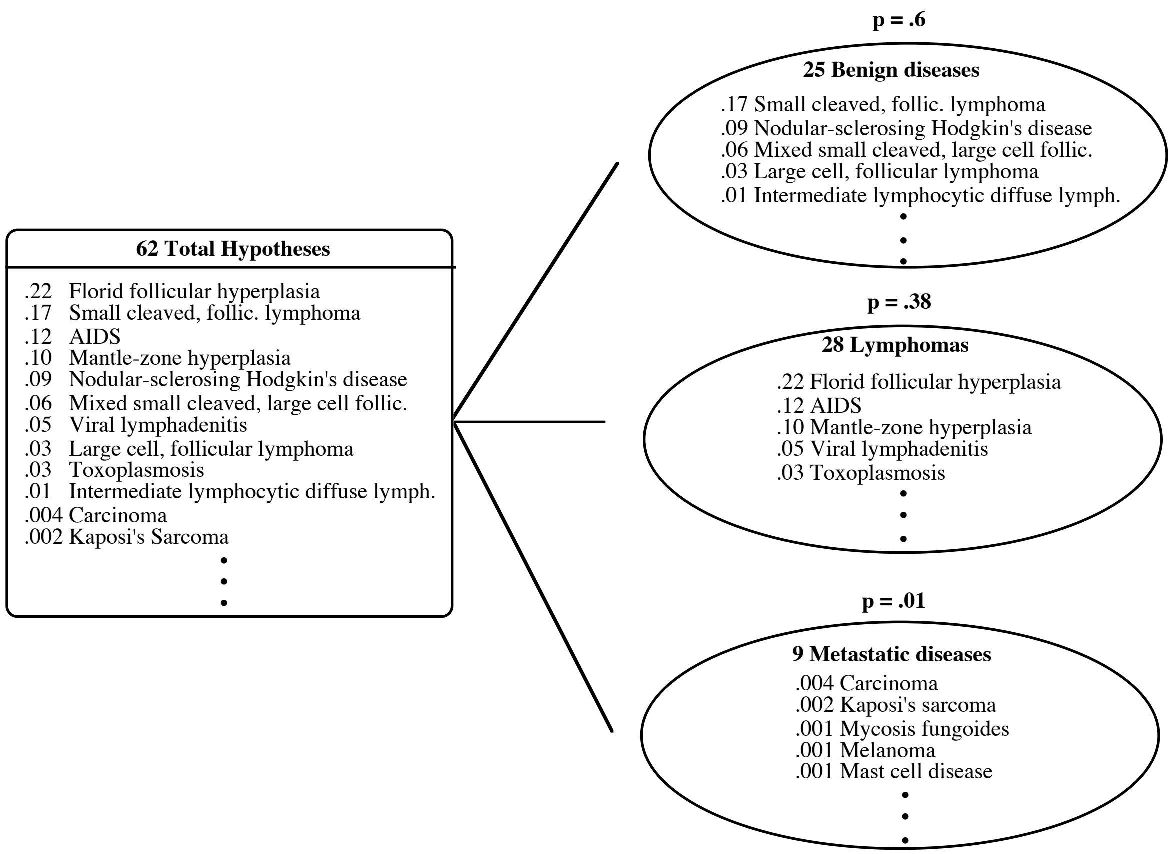


Figure 8.9: A Pathfinder abstraction.

This schematic displays a portion of a Pathfinder differential diagnosis, listing all diseases under consideration by likelihood. By invoking a simple etiological abstraction, we can divide the differential diagnosis into major disease classes, and assign the classes probabilities. A value-of-information analysis can consider these classes as single distinctions.

explanation of Pathfinder recommendations.

Figure 8.10 contains a Pathfinder screen demonstrating the abstraction capability. Here, the diagnostic problem is being viewed from the *etiological*, *pattern*, and *leading disease versus other diseases* perspectives. In the latter case, the grouping strategy is based on the distribution of belief, rather than on properties of diseases. The probabilities of major groups are listed in each window. In the *pattern* window, the differential-diagnosis title bar tells us that there are 7 Hodgkin's and 2 nonHodgkin's lymphomas under consideration. The probability that the final disease will be a Hodgkin's lymphoma is 0.999. The probability that it will not is 0.001. In the *leading versus other* window, shown in the foreground, the leading disease, *Hodgkin's disease, nodular-sclerosing* (HDNS), is currently assigned a probability of 0.473. The other 8 diseases under consideration other have a probability of 0.527. Invoking the value-of-information analysis while this window is active, will find those tests and features that will rule out contenders for the leading disease. The system allows a user to compare easily questions generated by information-gathering recommendations from the perspective of an of the abstraction strategies. When new information becomes available, all windows are updated.

8.6 Simplifying Justification with Abstraction

We have combined the abstraction techniques with graphical justification to explain the discriminatory value of Pathfinder test recommendations. Pathfinder graphs the influence that reporting each value of a feature will have on the likelihood of two diseases or *two groups* of diseases. For two groups of diseases G_1 and G_2, the system generates a set of likelihood ratios for each possible value i of the observed test or feature, E:

$$\frac{p(E_i|G_1)}{p(E_i|G_2)}$$

where $p(E_i|G_j)$, $j = 1, 2$, is the probability that evidence E of value i is observed given that the true disease hypothesis is in group G_i. We display the logarithm of each likelihood ratio. The logarithm of the likelihood ratio is called the *weight of evidence* (Good, 1950). Several other investigators have experimented with the

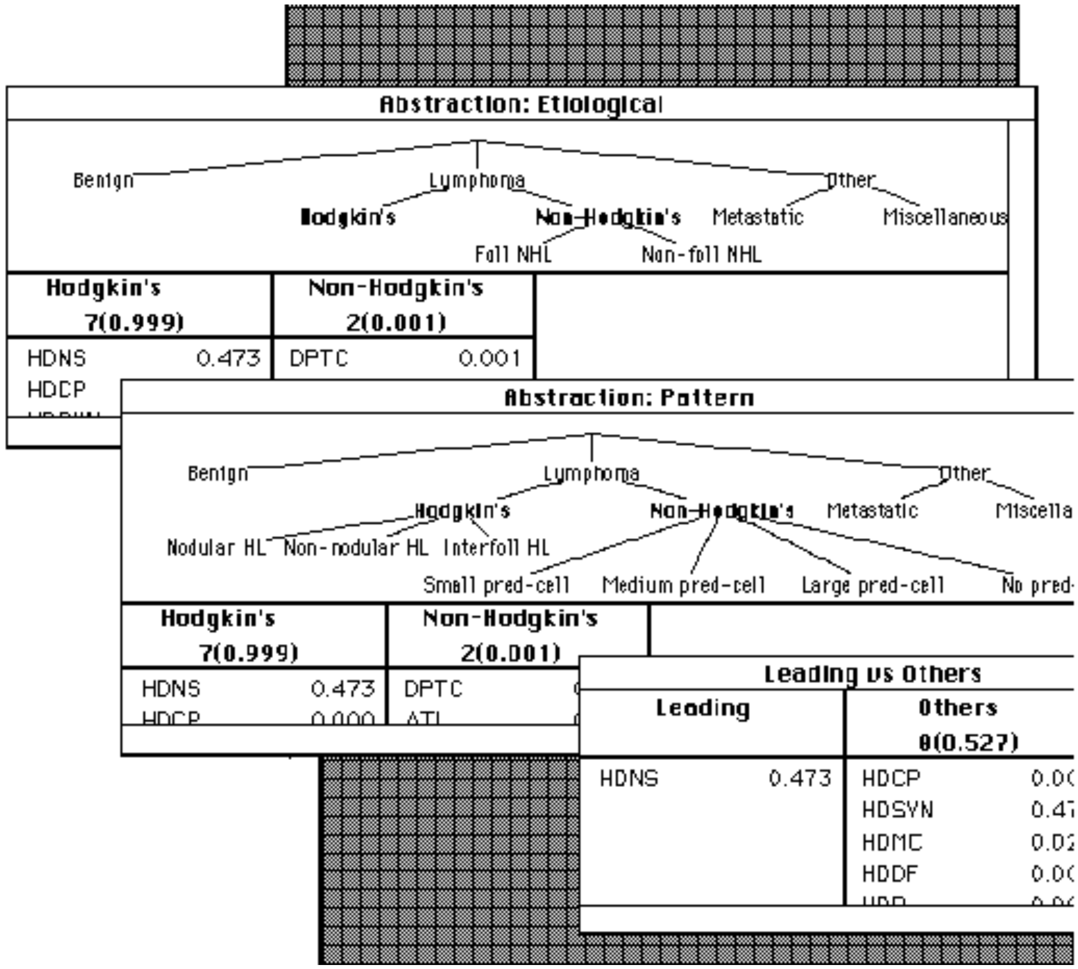


Figure 8.10: A Pathfinder screen.

This bitmap of a Pathfinder screen demonstrates the functioning of Protos/PF. Alternative abstraction hierarchies for managing the complexity of diagnostic inference are made available. A pathologist invokes the system to generate a recommendation tailored to a particular perspective by pointing with the mouse cursor at one of the windows. The currently active process, in the foreground, displays the problem from a *leading disease versus others* perspective.

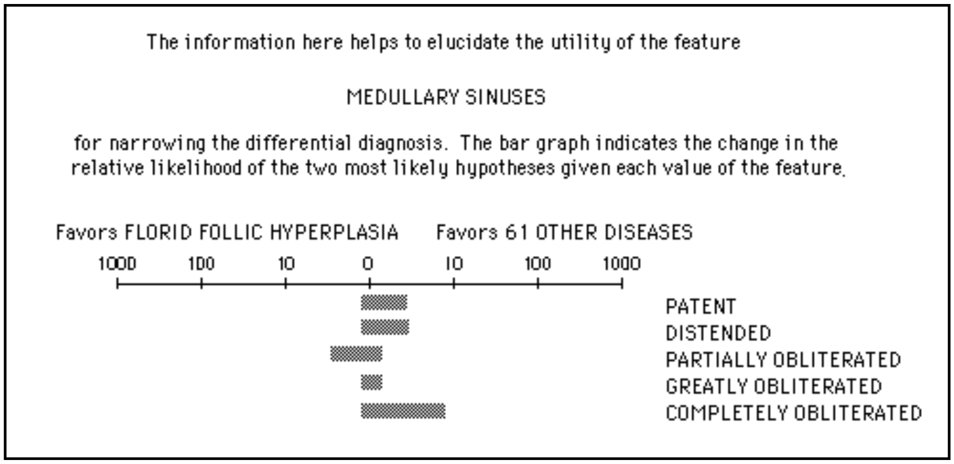


Figure 8.11: Pairwise justification.

This screen displays a sample explanation from Pathfinder, showing the use of weights of evidence to reason about the update that will be provided by different answers to the recommended feature, MEDULLARY SINUSES. In this case, we examine the problem from the perspective of *leading disease* versus *other diseases*.

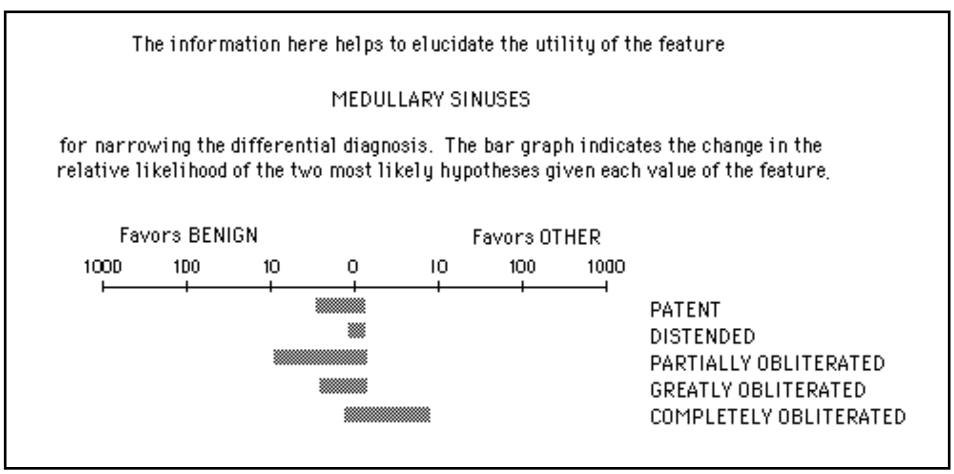


Figure 8.12: Value of information from another perspective.

By invoking a simple abstraction that distinguishes between *benign diseases* versus *other groups of diseases* (in this case, lymphomas and metastatic diseases), we can review the effect that different answers to the recommended feature, MEDULLARY SINUSES will have on the likelihood of these groups.

display of weights of evidence to explain the effect that a finding will have on belief in two diseases (Spiegelhalter and Knill-Jones, 1984; Reggia and Perricone, 1985). The Pathfinder work with abstraction is the first use of this approach to inspect the discriminatory power of evidence from multiple perspectives.

Figure 8.11 displays a sample Pathfinder justification of an observation, MEDULLARY SINUSES, that has been recommended by a value-of-information analysis on the diagnostic problem displayed in Figure 8.9. Figure 8.11 examines the problem from the perspective of *leading disease* versus *all other diseases*. Figure 8.12 shows how we can examine the effects of the same piece of evidence on a different pairwise abstraction. By invoking the abstraction that examines, *benign diseases* versus *other groups of diseases*, we can review the effect that alternative answers to a recommended observation will have on the likelihood of these groups. Figure 8.12 shows us that observing MEDULLARY SINUSES=*Patent*, is evidence against the leading disease, Florid Reactive Follicular Hyperplasia (FRFH). FRFH is a benign disease that typically resolves without therapeutic intervention. Figure 8.11 shows us that, from the perspective of a set of disease groups, the same observation (MEDULLARY SINUSES = *Patent*) is *evidence for* the benign group–the group containing FRFH.

8.7 Future Research on Abstraction

A promising extension of my work on increasing the clarity of reasoning through abstraction of distinctions in normative reasoning systems is the development of a normative metareasoner to control abstraction. In general, we wish to consider different levels of detail in reasoning about actions, states of the world, and observations. A normative control reasoner would seek to optimize the value of normative reasoning by trading off the informational costs associated with the use of an abstraction strategy with the benefits of simplicity and understandability of reasoning. Such metareasoning could make use of knowledge gathered by assessing a user's preferences about cognitive style, and using these preferences to make decisions about abstraction during the course of a diagnostic or an educational session. A discussion of the control of abstraction for the explanation of automated reasoning is found in

(Horvitz, 1987a). Related study of the utility-based control of explanation is also described in (Mclaughlin, 1987).

There is much room for doing more detailed assessment of preference about the detail, perspective, and progression of a normative analysis. During my exploration of pathologist's preferences about abstraction, I found different cognitive styles with regard to such factors as the cost of making a transition from one perspective, or level of abstraction, to another, and the cost of making a transition from one class of information to another (e.g., being asked by the system to move between the evaluation of microscopic features at low- and high-power objectives). Another promising area of research is the refinement of the notion of a decision-theoretic perspective, not only to capture preferences about problem abstraction, but also to account for different dimensions of patient utility. For example, a physician may wish to compare the differences between diagnostic recommendations, given the monetary constraints on the tests that are currently available at an institution, with recommendations that would be ideal for a less constrained analysis.

8.8 Summary

The complexity of decision-theoretic inference and the inflexibility of normative diagnostic systems has bolstered the stereotype of such systems as being necessarily rigid and unnatural. Perceptions about the opacity of normative reasoning stem in part from an inability to address human cognitive-resource constraints. I described abstraction techniques that allow us to represent and use heuristic, human-oriented abstractions within a decision-theoretic system. This new capability increases the naturalness of normative inference and explanation. We have found the graceful integration of such flexibility to be useful in adapting computer-based inference to clinicians. Considering the constraints imposed on detailed decision-theoretic inference by human cognition can motivate us to enrich the cognitive style of normative reasoning systems by generating flexible and human-oriented approximation strategies. Such strategies can help to make automated decision-theoretic reasoning more compatible with people.

Chapter 9

Summary of Contributions and Future Research

In this dissertation, we explored flexible computation strategies and developed and examined decision-theoretic techniques for directing problem solving. We described how the methods could be used to optimize the value of computation under varying resource constraints. The decision-theoretic control techniques take into consideration information about the expected costs and benefits of continuing to refine a partial result versus those of taking action in the world. We used the techniques to develop a normative model of rational action for computer-based decision making under bounded resources. This model of rationality is founded on extending traditional decision-theoretic inference with efficient deliberative metalevel analyses that select the best solution procedure and ideal time to perform inference with that procedure. Such *normative metareasoning* has been reduced to practice in the Protos system. We shall review, in this chapter, the contributions my research makes to DA, AI, theoretical computer science, and medical informatics.

9.1 Contributions to Decision Analysis

From the perspective of DA, my research on normative metareasoning introduces a formal level of reflection to decision analyses. Traditionally, decision analysts have made heuristic decisions about the level of detail to represent in models and the precision with which to draw conclusions from them. This dissertation discusses procedures for making decisions *about* the nature and extent of a decision analysis. These normative-metareasoning techniques can enable a human or computer-based reasoner to use incomplete or uncertain characterizations of the value of continuing compute to select an analysis with the highest expected utility. Our formal handling of resource issues does not replace heuristic reasoning with axiomatic principles; many decisions about the detail and nature of the metalevel and object-level decision models are derived through human intuition. However, normative metareasoning *extends the frontier of formal normative analyses* to include a consideration of resource constraints.

Beyond the use of reflective decision analyses to select ideal inference policies under time constraints, we also examined normative reasoning under cognitive constraints. This research highlights the significance of simplifying normative inference for the purpose of making decision-theoretic reasoning easier to explain and understand. The research shares with the work on time-constrained reasoning a consideration of the tradeoff between the completeness of an analysis and the value of that analysis. In the case of cognitive-resource considerations, I found that increasing the level of abstraction and, thus, reducing the precision of a decision analysis, could increase the value of the analysis to a user because of gains in the explainability of the simpler analyses.

Finally, I have worked in my dissertation research to bring difficult problems of autonomous decision making posed by AI investigators to the attention of the DA community. To date, challenging automated reasoning problems have remained largely within AI. Nevertheless, the problems of normative reasoning under resource constraints pose rich offline and real-time computational decision problems that promise to interest many decision-science investigators.

9.2 Contributions to Artificial Intelligence

The design of intelligent artifacts that can perform well in complex environments, given constraints in computational and memory resources, has been a central challenge in AI research (Simon, 1969). My dissertation research addresses the formal foundations of rational action by autonomous decision-making agents under bounded resources. I introduced the notion of bounded optimality as distinct from the more general notion of bounded rationality in AI. In pursuing bounded optimality, we seek to develop principles of rational action based on offline or run-time considerations of the costs and benefits of alternative computational strategies for object-level and metalevel reasoning, and on the utilities and probabilities of outcomes. Rather than reject normative methods as inappropriately complex, I have sought to use decision-theoretic principles to select the degree of detail in object-level decision-theoretic analyses. I have shown how such optimization can increase the value of problem-solving systems in dynamic environments.

I discussed the use of tractable decision-theoretic tools as a promising means of addressing the complexity of base-level decision-theoretic inference, and of custom-tailoring the completeness of problem solving dynamically, in response to changes in the cost of computation. I posed normative-metareasoning—the use of decision analysis to make offline and real-time decisions about the configuration and reasoning policies for reasoning systems—as the key to developing systems that are rational under limited computational abilities and resources. In particular, *reflective decision analysis*—the use of normative metareasoning to control probabilistic inference in a decision context—promises to be useful in addressing a variety of AI challenges, including problems with planning and search.

Beyond the introduction of normative metareasoning and reflective decision analysis, my research makes contributions to AI research in the following areas:

- *Flexible computation.* I described how the properties of flexibility—monotonicity in the refinement of one or more attributes of a partial result with computation and convergence of the attributes of a partial result on a final result—can be valuable in the economic optimization of the value of computation under uncertain and varying resource constraints. The pursuit of flexible problem-solving procedures promises to yield robust strategies under varying resource constraints for a wide variety of problem-solving tasks.

- *Protos reflective decision system.* The decision-theoretic Protos architecture is an example of a useful configuration for intelligent reasoning systems. Highlights of the Protos architecture include object-level computation strategies and a decision-theoretic metalevel reasoner that applies several forms of metareasoning to analyze the value of computation. The metalevel analyses allow the system to respond to a time-dependent challenge with a reflex or a more deliberative inference policy.

- *Formal perspective on control.* AI investigators have traditionally relied on heuristic procedures for choosing among different reasoning strategies (Erman et al., 1980; Hayes-Roth, 1985; Clancey, 1985; Cohen et al., 1989; Durfee and Lesser, 1987). In addition to giving us a framework for the design and optimization of computational behavior, normative metareasoning provides a formal foundation for the control of reasoning in AI applications. The principles of normative metareasoning have promise for guiding reasoning in applications where heuristic control has been used.

- *Decision-theoretic approach to real-time reasoning.* My research contributes a principled perspective on reasoning under real-time constraints—an area of growing interest in the AI community. The general trend of research in real-time reasoning is to build systems that react quickly with a set of logical reflex, or *situation–action* rules (Rosenschein and Kaelbling, 1986; Agre and Chapman, 1987; Brooks, 1987; Kaelbling, 1987). As viewed from the normative metareasoning perspective, the techniques for generating such rules typically have been suboptimal as investigators have relied on ad hoc models. As normative reasoning has been viewed as a source—rather than a solution to—problems of intractability, decision-theoretic methods generally have been dismissed as a solution to real-time reasoning (Laffey et al., 1988). There is promise in using normative meta-analyses for designing inexpensive real-time strategies.

Decision-theory–based reasoning systems, resulting from such analyses, might employ explicit real-time analyses, or use only rules and policies generated in detailed offline analyses.

9.3 Contributions to Computer Science

From the perspective of theoretical computer science, my research introduces decision-theoretic foundations for optimizing approximate calculations in different contexts. Rational decisions about computation, such as the selection of a new strategy or the decision to cease computing, can be sensitive to details of that strategy's refinement trajectories, to the object-level utility function, and to the uncertainties in the functions describing the cost and availability of reasoning resources. A wide range of computer-science research efforts on fundamental computational problems may benefit by pursuing the development of reflective strategies that are sensitive to varying resource and utility conditions.

The framework and examples presented in this dissertation highlight the opportunity for developing and characterizing alternative approximation processes that can refine multiple attributes of partial results with computation. Formal research could augment complexity-class results with proofs of worst-case rates of incremental refinement, theorems about ideal problem-solving policies, and analyses of fundamental tradeoffs in how algorithms refine different attributes of results. Multiattribute analysis, flexible computation, and decision-theoretic control promise to be especially useful in constructing and characterizing ideal approximate solutions for intractable problems in the $\mathcal{NP}$-complete[1] complexity class.

9.4 Contributions to Medical Informatics

The evolving discipline of medical informatics has strong ties to DA and AI. Medical information scientists have diligently applied and refined techniques developed

[1] $\mathcal{NP}$-*complete* refers to the fact that a theoretical construct, called a *nondeterministic computational automata*, requires an amount of time to solve a problem that grows polynomially with the size of that problem. Problems in this class include the task of finding an optimal solution to the traveling-salesman problem (TSP), and many other problems (Garey and Johnson, 1979).

in each discipline. Indeed, many medical-informatics investigators have continued to straddle the AI–DA border. Researchers interested in building computer-based reasoning systems that can improve medical decision making have been attracted to the expressiveness and tractability of heuristic reasoning techniques. Nevertheless, many of the same investigators are attracted to the mathematical elegance and principled foundations of probability and decision theory. For many of the researchers, decision theory has served as a gold-standard for medical decision making. However, experience with the inflexibility and complexity of traditional approaches to decision-theoretic reasoning and representation have given investigators reason to associate the normative approach with intractable computation and opaque reasoning.

Normative metareasoning and flexible computation can help bridge the conceptual gap between normative, approximate, and heuristic reasoning—and can extend the applicability of principled, normative systems. Flexible reasoning methods, coupled with decision-theoretic control, provides a principled foundation for building reasoning systems that can custom-tailor the detail and level of normative analyses to specific medical challenges and users. My work on normative metareasoning was partly motivated by the high stakes, uncertainty, and time criticality of medical decision making. Extensions of techniques developed in this dissertation may be valuable for building physician advisory tools, and for developing real-time decision making and control mechanisms for monitoring and therapy equipment. In particular, normative metareasoning techniques promise to play a crucial role in the development of effective reasoning systems that can draw conclusions from very large belief networks. For example, there is great opportunity for the use of flexible reasoning and normative metareasoning in the ongoing QMR-DT project (Shwe et al., 1990a; Shwe et al., 1990b). The evolving QMR-DT belief network represents diseases and manifestations spanning the broad domain of internal medicine.

9.5 Research Challenges and Opportunities

Many interesting problems of computation and action under bounded resources remain largely unsolved. I shall describe several promising challenges and opportunities for extending the research presented in this dissertation.

9.5.1 Consideration of New Information

The current implementation of Protos assumes that a set of observations about the state of the world becomes available at intervals of time that make consideration of information acquisition irrelevant during deliberation. This assumption may be invalid. In the general case, an agent should integrate a consideration of decisions about the opportunity to exchange some valuable commodity for additional information with metareasoning about the value of computation. For example, delay for computation can be synergistic with the availability of new information. Also, new information that might become available during deliberation can indicate a change in one or more time-dependent utility functions, or, more generally, in the decision model. Thus, value-of-information considerations should be considered with value of computation in more general models. In general we should consider the value of information and the value of computation, conditioned on gathering new information. We can assume separate deliberation processes for gathering information and for thinking about the information. On the other hand, we can reason about whether it is more useful to reason about the gathering of new information versus computing with the information already observed.

9.5.2 Automatic Construction of Decision Models

Protos currently assumes that the distinctions in its decision model will remain unchanged during deliberation about action. We reviewed the difficulties with the automated *framing* or formulation of decision problems in Chapter 4. Although we have few computer-based tools for problem formulation, the development of methods for the real-time construction of decision models promises to be a crucial component of autonomous computer-based reasoners that take action under limited and uncertain resources. Research to date on problem formulation (Holtzman, 1989; Breese, 1990; Wellman, 1988; Heckerman and Horvitz, 1990) presents opportunities for extension to systems that can formulate dynamically decision problems for solution in bounded-resource reasoners. A challenging area of research is the use of normative metareasoning principles to reason about the costs and benefits of constructing and continuing to refine a decision model, given goals and observations.

9.5.3 More Sophisticated EVC Analyses

In the Protos implementation, we abstract relevant states of the world into a set of decision problems that depend on refining one or a small number of probabilities. Increasing the number of uncertain states of the world leads to a growth of complexity of the EVC analysis. We discussed how we can use the inexpensive EVC to iteratively examine a large set of decision problems, and for reformulating a complex set of decisions into a cascade of simpler analyses. Nevertheless, there is promise for developing more sophisticated approximations for EVC, to make feasible the analysis of ideal action given a larger number of uncertain hypotheses about the world.

9.5.4 Compilation of Reasoning and Metareasoning

There is opportunity for developing reasoners that make use of prestored or *compiled* actions, instead of computing action—or meta-action—in real time. Several different classes of compiled knowledge for reasoning and metareasoning are discussed in (Horvitz, 1989a). We can precompute and store complete answers or actions in the form of situation–action pairs that describe ideal actions to take, given a set of observations. Beyond storing final actions, we can presolve and store incomplete solutions that can be extended to a specific solution at run time. For example, rather than perform real-time probabilistic inference, we can compute a set of probabilities offline and store them for use in real-time decision making. We might also store a set of instance weights for use in bounded conditioning (as discussed in Section 5.4.1). We refer to such knowledge as *platform* knowledge, as it involves giving a reasoning system a headstart through reducing the overhead in real-time at the expense of memory. Finally, we can identify and store compiled *resource rules* that provide a means of buying time for computation. An example of a resource rule in medicine is the reflex maneuver of continuing to replace fluids in a trauma patient to maintain blood pressure while the source of a hemmorhage is being investigated.

Preliminary research on the development of decision-theoretic rules for designing optimal sets of compiled situation–action rules in an offline setting has been reported by Heckerman and colleagues (Heckerman et al., 1989a). In that work, the value of using completely deliberative reasoners is compared to the use of completely compiled models. In another study, Herskovits and Cooper show how to decrease the average response time to probabilistic queries by building a tree of cached probabilities from a belief network ahead of time (Herskovits and Cooper, 1989). These investigators demonstrate how to select cases for caching by performing simulation in the network weighted by utility. The optimization of the construction and use of such trees, given limitations in memory and time, are open areas of research.

9.5.5 Integration of Compiled and Deliberative Actions

Developing techniques and general architectures for utilizing a spectrum of default and precomputed results promises to be valuable. In several places in this dissertation, I touched on interrelationships between the dynamic computation of results, on the one hand, and the use of compiled responses, on the other. I described the use of a default metareasoning policy for performing a preliminary EVC evaluation in Protos to allow the system to take a reflex action based on the mean of the current belief, rather than be forced to perform a complete meta-analyisis. The construction of a deliberative normative metareasoner can make clear the expected savings in the utility of storing a compiled result versus performing inference. Thus, a normative metareasoning architecture can serve as a basis for selecting a set of compiled actions to be used in conjunction with the deliberative reasoner.

Just as we can develop compiled–deliberative object-level reasoners, we can do the same with normative metareasoners. We can also combine such integrated metareasoners with integrated object-level reasoners. Such a combined approach could generate a variety of appropriate reflex and deliberative actions. Finally, beyond design-time compilation, there is opportunity for developing techniques to use idle time effectively; offline normative meta-analyses might show that, for many contexts, ideal agents should be anxiety-ridden reasoners that worry about and plan for expected problems, much as people do.

9.5.6 Utility-Directed Program Synthesis

By focusing decision-theoretic metareasoning procedures on the microstructure of computational activity, we may generate patterns of computation and control possible may enable a reasoner to generate refinement trajectories that provide more valuable computation than manu of our current algorithms. Costs and benefits under uncertainty. For example, a decision-theoretic paradigm for controlling divide and conquer methods typically allocate resource for (1) decomposing a problem instance into a set of subproblems, (2) solving the subproblems, and (3) recombining the subproblem solutions into a final answer. The more subproblems we produce, the smaller and more tractable each subproblem becomes; however, as we decrease the size of the subproblems, we increase the number of subproblems. We also increase the costs of decomposition and assembly of the subproblem. The ideal decomposition of a problem depends on the tradeoffs between the complexity of decomposition and subproblem solution. Structural control techniques might be employed to optimize the value of a divide-and-conquer algorithm, under the general condition of uncertainty in the costs of decomposition and subproblem solution.

Such research may also elucidate the control strategies implicit in familiar policies and stimulate the creation of more general, decision-theoretic strategies that could implement the familiar policies as special cases. The study of fine-grained approaches to decision-theoretic inference can also elucidate methods for reasoning about utility in influence diagrams directly so that attention can be focused on the most relevant parts of a problem, as opposed to inferring the probabilities of alternative outcomes with algorithms for probabilistic-inference.

9.5.7 Preference Models for Histories of Action

In Chapter 7, we examined the behavior of Protos for test cases involving the use of different belief networks and time-dependent utility models for making one-shot decisions. We would like to evaluate the value of normative-metareasoning systems, such as Protos, for addressing a large number of challenges over time in some environment. The EVMP—the value of a reasoning policy for addressing a population of challenges—is a measure that captures the value of an agent over time. Computing the EVMP requires models for representing a complex lottery of large sets of actions and outcomes over time. Unfortunately, to date, decision-science investigators have provided few tools for evaluating complex histories of actions taken in response to challenges. There is a need for developing functions for combining the outcomes, generated by an agent's responses to challenges over time, into a measure of preference. Such preference models for histories of actions would allow us to compare the relative value of immersing different reasoning systems in a partially characterized environment. One approach to grappling with multiple actions and outcomes is to assume independence among distinct challenge-action pairs. I have previously posed the *independent-challenge model* as an approximation model for this task (Horvitz, 1987c). Unfortunately, the independence assumptions of this model are often invalid.

9.5.8 Analyses of Heuristic and Descriptive Strategies

In the work on Protos, I applied principled metareasoning to control well-characterized approximation strategies at the object-level. We can apply similar techniques to determine the expected value of heuristics. Extending normative metareasoning to consider the performance of heuristics, in addition to better-characterized normative approximations, would require us to consider uncertainty in the performance of heuristic strategies. We can gather information about the uncertain performance of heuristic procedures with statistical analyses of problem-solving behavior for different problem instances. Preliminary characterizations of heuristic behavior could be refined with ongoing data collection. A normative-metareasoning perspective could provide a synthesis of normative and heuristic procedures, even when we are limited to poor characterizations of heuristics.

Also, there is opportunity to develop decision-theoretic analyses to demonstrate the relationship between heuristic and rigorous bounded-resource analyses of optimal belief and action. For example, normative metareasoning may offer a new perspective on descriptive models that characterize human decision making. Investigators have shown that people exhibit stereotypical deviations from the axioms of utility theory, referred to as *biases* of judgment and decision making (Tversky and Kahneman, 1974; Kahneman et al., 1982). Normative meta-analyses may help us to characterize the suboptimality of nonnormative behaviors exhibited by people. We may find that, in some cases, heuristics are approximate solutions to normative-metareasoning problems of action under bounded resources.

Normative metareasoning may have relevance to the study of neural network models of problem solving, and, more speculatively, to natural reasoning systems. Experiments that demonstrate the value of metareasoning and control in computer-based models suggest that such facilities may play a role in natural reasoning systems. It is feasible that approximations of normative metareasoning may well play an important role in the cognitive systems of living creatures developed through long-term optimization under the pressures of competition. Given competition for limited resources and cognitive systems bounded resources The pressures of natural selection could have Metareasoning and control facilities in natural cognitive systems could have been selected under the pressures of evolution. The reflective component of deliberative reasoning in people as something that raises the expected value as a feature that natural selection. The perspective gained might suggest useful directions in examing the architecture of natural cognitive systems. may have discovered the value of allocating a portition of cognitive resources to an explicit metalevel of analysis. use of uncertain information about problem solving given constraints on the architecture of a reasoning system, and on the costs of memory and inference.

9.5.9 Ideal Partition of Resources for Metareasoning

In this dissertation, we focused on the development and use of tractable metareasoning policies to control object-level reasoning. We did not dwell on the control of metareasoning with metametareasoning techniques. However, as we mentioned in Chapter 2, we may not always be able to simplify the metareasoning problem to a single inexpensive analysis of the value of computation. In some situations, we may have flexible—or a set of alternate—metareasoning approaches. In some situations, the most valuable metareasoning strategies may use a large fraction of the total resources consumed by problem solving. In other cases, the most valuable strategy may use very little of the total resources. Thus, we may wish to reason about the ideal control of metareasoning. An important component of metametareasoning is defined by the *metareasoning-partition* problem—ideally apportioning costly reasoning resources to metalevel inference versus applying resource to executing a solution to a problem. Insights about the ideal partition of resources for metareasoning in several prototypical resource contexts have been explored recently in (Horvitz and Breese, 1990) and (Breese and Horvitz, 1990).

9.5.10 Addressing Problems of Analytic Regress

I discussed, in Section 2.6, how working to enhance the value of object-level computation, by introducing one or more levels of metareasoning and control, does not necessitate intractable metareasoning or infinite analytic regress. Concerns with analytic regress that arise in discussions on the value of adding a single metalevel result often from an assumption that any metareasoning implies a need for the recursive application of meta-analysis. Such concerns can be provoked by an assumption that a metareasoner must represent a great portion of the base problem that it is intended to control and, thus, must be as complex as a base-level problem solver. Therefore, investigators may argue, a metalevel reasoner will require the same kind of control as that needed by the base-level reasoner.

We may, indeed, be able to construct metaproblems with a complexity that rivals or exceeds that of the base problem. However, we can frequently build much simpler control problems that greatly enhance the comprehensive value of reasoners. Such meta-analyses address particular aspects of object-level problem solving, and make use of specific classes of metaknowledge; they do not capture the full complexity of the representation or inference procedures at the base level.

Another assumption that can provoke concern about infinite regress is that control decisions can be sensitive to small changes in the accuracy of a metareasoning analysis. If this were so, optimizing the value of control decisions might invoke a large, or an infinite, number of metalevels. I believe that such sensitivity is the exception, rather than the rule. Within the models that I constructed, control decisions appear to be insensitive to small changes in the accuracy of the analysis. When this is not the case, we can seek to characterize a relevant spectrum of metametareasoning implications during the design phase.

Questions about infinite regress grow in significance if we seek to verify the absolute optimality of a decision procedure, especially if metalevels can be of arbitrary complexity. There can be cases where the composition of an agent and the agent's environment dictates as optimal the application and management of a large, or (in theory) infinite, tower of meta-analyses (Horvitz, 1988). In previous discussions of techniques for handling the problems with analytic regress, Kripke presented a problem with infinite regress that arises in defining truth (Kripke, 1975). He applied a mathematical tool known as a *transfinite hierarchy* to grapple with the problem. More recently, Lipman performed an analysis of an infinite-regress problem by applying similar techniques within a game-theoretic framework (Lipman, 1989). He demonstrated that, within his model, there is an equivalence between a base decision problem and an infinitely recursive analysis.

From the normative-metareasoning perspective, optimization problems that imply large hierarchies of metareasoners provide problems for convergence and stability analyses. Nonetheless, analyses may suggest finite architectures and policies, even for an ideal bounded-optimal agent that relies on theoretically unwieldy, recursive meta-analyses. The EVC of a reasoning system, based on a theoretical prescription for an infinite hierarchy of metareasoning analyses, might converge tractably, similar to the way the sum of an infinite series may closely approach a real number with a small number of terms. Also, stability analyses, developed by control theorists, may be useful in determining optimal configurations of large numbers of interrelated meta-analyses, especially for cases where some control decisions are sensitive to minute perturbations of the accuracy of the analysis at a large number of metalevels. Great sensitivity of control reasoning to the completeness of metareasoning analyses can focus attention on a search for less sensitive, more stable levels of approximate meta-analyses.

Appendix A

Decision Theory and Decision Analysis

This appendix provides background on probability, decision theory, and decision analysis. After reviewing the key notion of probability as personal belief, the axioms of probability, and the axioms of utility, I shall present the belief network and influence diagram representations. Then, I shall describe research on algorithms for performing inference with these representations.

A.1 Probability and Decision Theory

Systems that reason about problems in the real world typically can consider only a small fraction of potentially relevant distinctions. Thus, most representations used in automated reasoning systems are dramatic simplifications of the objects and relations in the universe that may have relevance to a decision problem. Incompleteness in the representation of distinctions in the world, and in the state of information about distinctions that are represented, leads to uncertainties about different outcomes and about the consequences of actions. Thus, computer-based reasoning systems—as well as people–must contend generally with uncertainty.

The development of modern probability theory as a set of axioms and as a formal calculus for representing and reasoning about partial truth, or *belief*, has its roots in the writings of Pascal, Fermat, and Bernoulli in the sixteenth and seventeenth centuries. However, detailed discussions of uncertain reasoning appear much earlier in history. Informal attempts by people to grapple with the ubiquity of uncertainty have appeared throughout history. For example, discussion about uncertainty and its relationship to games of chance appear many hundreds of years ago in Egyptian, Greek, and Indian literature (Hacking, 1975). Games of chance were employed in some cultures as a means of discovering the will of the gods; games of chance, such as observing the way a cuboidal bone would land when tossed in the air, were viewed as providing gods a means to communicate with mortals (David, 1962).

Probability theory serves as a language for making statements about uncertainty; the theory makes explicit the notion of partial truth and incomplete information. Utility theory is a set of axioms, some of which are based on probability, that provides a language for reasoning about the value of alternate actions under uncertainty. Analyses by Pascal and others identified the notion of expected value several centuries ago. However, von Neumann and Morgenstern formalized utility theory only four decades ago (von Neumann and Morgenstern, 1947). Probability theory and decision theory provide principles for rational beliefs and actions under uncertainty. The theoretical concepts do not, in themselves, provide information about how to best apply these principles to real problems in an efficient manner. Investigators in the discipline of em decision analysis have explored the real-world application of decision theory, often making use of heuristics about the *process* of decision modeling, information acquisition, and action in the world.

A.1.1 Probability as Personal Belief

The most widespread interpretation of probability is that of a measurable frequency of events determined from repeated experiments. From this perspective, we define the probability that a coin will land on one of its sides, given a random flip, as the portion of times the coin lands on that side, given some large number of coin flips. This view has been called the *frequentist* interpretation. A different perspective on probabilities, known as the *subjectivist* interpretation, is that probabilities are a measure of a person's degree of belief in the event, given the information available. According to this conception, probabilities refer to a *state of knowledge* held by an individual rather than to properties of a sequence of events. A probability of 1 corresponds to belief in the absolute truth of a proposition, a probability of zero to belief in the proposition's negation, and intervening values to partial belief. Subjectivists are also called *Bayesians*, referring to the work of Thomas Bayes, a theologian and scientist from the mid-eighteenth century, who provided one of the earliest discussions of probability as a personal measure of belief (Bayes, 1958).

Subjective probabilities abide by the same set of axioms as do classical probabilities or frequencies. The subjectivist approach is a generalization of the more popular notion of a probability as a long-run frequency of a "repeatable" event. In addition to assigning probabilities to the outcomes of repeatable events, a subjectivist is willing to assign belief to unique events that are not members necessarily of any obvious repeatable sequence of events. The assignment of a subjective probability is based on all information available to an individual. Such background information includes those items that are known to be true or deducible in a logical sense, as well as empirical frequency information.

As an explicit statement that a probability is a personal belief that can include a consideration of information that may not be expressed explicitly, Bayesians frequently include a reference to a state of *background information*, ξ, on the which the probability is based or *conditioned*. This symbol is included in the conditioning clause of probabilities, $p(Q|\xi)$, and is concatenated with new explicit observations or evidence E, $p(Q|E, \xi)$.

A.1.2 Axioms of Probability Theory

The axioms of probability define probability and provide a set of rules for the consistent combination of the probabilities of related events. There have been several equivalent axiomitizations of probability. We shall use the following formulation, where Q and R are different events:

$$0 \leq p(Q|\xi) \leq 1 \tag{A.1}$$

$$p(Q \text{ or } \neg Q|\xi) = 1 \tag{A.2}$$

$$p(Q|\xi) + p(\neg Q|\xi) = 1 \tag{A.3}$$

$$p(Q, R|\xi) = p(Q|R, \xi)p(R|\xi) \tag{A.4}$$

Equation A.1 defines the measure of probability as extending between 0 and 1. Equation A.2 defines the probability of the disjunction of an event and its negation as true. Equation A.3 tells us that the sum of the probability of an event and its negation is 1. This axiom is referred to as the *sum rule*. A list of propositions, separated by commas, is used typically to denote logical conjunction. Equation A.4 tells us that the probability of the conjunction of two events is equal to the probability of one event, given the truth of the second event, multiplied by the probability of the second event. This axiom is called the *product rule*.

Sets of belief assignments that are consistent with the axioms of probability theory are said to be *coherent*. In this sense, the axioms provide consistency criteria for uncertain beliefs. There are persuasive arguments for combining beliefs in a manner that is consistent with probability theory. Decision analysts have presented examples of scenarios that suggest that a rational person would wish to avoid making decisions based on incoherent beliefs (Lehman, 1955; Shimony, 1955; de Finetti, 1970).

In another vein of research, investigators have provided sets of desirable properties of measures of belief that they consider more intuitive than the axioms of probability (Cox, 1946; Tribus, 1969; Lindley, 1982). In 1946, Cox provided a proof that a set of such properties are logically equivalent to the axioms of probability. A recent reformulation of Cox's properties is as follows (Horvitz et al., 1986a):

1. *Clarity*: Events should be well defined.

2. *Scalar continuity*: A single real number is both necessary and sufficient for representing a degree of belief in an event.

3. *Completeness*: A person can assign a measure of belief to any well-defined event.

4. *Context dependency*: The degree of belief assigned to an event can depend on the belief assigned to other events.

5. *Hypothetical conditioning* : There exists some function f that allows the belief b in a conjunction of events, $b(x, y)$, to be calculated from the belief in one event and the belief in the other event, given that the first event is true. That is,

$$b(x, y) = f(b(x|y), b(y))$$

6. *Complementarity* : The belief in the negation of an event is a monotonically decreasing function of the belief in the event; thus, the belief in an event's negation can be computed from the degree of belief in the event.

7. *Consistency* : There will be equal belief in events that are logically equivalent.

Cox demonstrated that, taken together, these properties logically imply that the measure of belief must satisfy the axioms of probability theory. According to Cox's analysis, if one accepts these intuitive properties as desirable, one must accept probabilities as a desirable measure of belief. The proof of the necessary relationship between the intuitive properties and the axioms of probability theory is based on an analysis of solutions to the functional forms implied by the intuitive properties.

A.1.3 Performing Inference Under Uncertainty

Probability theory allows us to reason about the degree of belief in events, given belief in related events. Bayes' rule is a useful relationship that allows us to calculate the belief in different hypotheses H about the world, $p(H|E, \xi)$, given observable evidence E, from knowledge about the probability of observing the evidence when each hypothesis is true, $p(E|H, \xi)$. The rule provides a method for transforming a prior probability $[p(H|\xi)]$ into a posterior probability $[p(H|E,\xi)]$ as follows:

$$p(H|E,\xi) = \frac{p(E|H,\xi)\ p(H|\xi)}{p(E|\xi)} \tag{A.5}$$

Bayes' rule can be derived in a straightforward manner from the sum and product rules of probability. The rule provides probabilistic inference with a bidirectionality that can be useful when probabilities are available in one direction but are required in the reverse direction. For example, in medicine, an expert physician may feel comfortable in assessing knowledge in terms of the likelihood of seeing alternate test results when different diseases are present, and wish to use this knowledge to compute the probability of a disease given one or more test results.

A.1.4 Axioms of Utility Theory

Decision theory is defined by the axioms of probability and utility. The axioms of utility, introduced by von Neumann and Morgenstern, are a set of principles for defining utility and for maintaining consistency among preferences (von Neumann and Morgenstern, 1947; Savage, 1972; de Finetti, 1970; Fishburn, 1981). An action or *decision* is an irrevocable allocation of resources. *Preferences* describe a person's valuation and ordering of alternate outcomes.

In the general, we must make decisions under uncertainty. A well-known rule for making decisions about action given alternative uncertain outcomes is the *principle of maximum expected utility* (MEU). The MEU principle dictates that we should take actions that maximize the value computed by summing together the value attributed to each possible outcome multiplied by the probability of that outcome. According to Hacking, Pascal was the first to specify the principal of maximum expected utility over 3 centuries ago in deliberating about the costs and benefits of practicing Christianity, given uncertainty about the existence of heaven and hell (Hacking, 1975). von Neumann and Morgenstern were the first to formalize MEU.

The axioms of utility provide justification of the principle of maximum expected utility (MEU). The axiom of *orderability* asserts that all outcomes are comparable, even if they are described by many attributes. Thus, for any two possible outcomes a and b, a person either prefers a to b ($a \succ b$), prefers b to a ($b \succ a$), or is indifferent between the two outcomes ($a \sim b$). The *transitivity* axiom asserts that these orderings can be assigned consistently; that is, if $a \succ b$ and $b \succ c$, then $a \succ c$. These axioms, together with two auxiliary axioms, ensure a weak preference ordering on all outcomes. This implies the existence of a scalar value function which maps from all outcomes into a scalar value such that a person will always prefer outcomes assigned a greater value.

The other axioms of utility theory describe a person's preferences under uncertainty. They make use of the concept of a *lottery*—a set of uncertain outcomes associated with an action. Each outcome in a lottery has some probability of occurrence. The *monotonicity* axiom states that, when comparing two lotteries, each with the same two outcomes but with different probabilities on the outcomes, a decision maker should prefer the lottery that has the higher probability of the preferred outcome. The *decomposability* axiom asserts that we can combine a set of lotteries into simpler lotteries and that a decision maker should be indifferent between lotteries that have the same set of eventual outcomes and probabilities—even if they are reached in a different manner. As an example, a lottery whose outcomes are other lotteries can be decomposed into an equivalent one-stage lottery using the standard rules of probability. The *substitutability* axiom states that, if a decision maker is indifferent between a lottery and some certain outcome (a *certain equivalent* of the lottery), then substituting one for the other as a possible outcome in some more complex lottery should not affect his preference for the lottery. The *continuity* axiom dictates that, if a person has preferences $a \succ b$ and $b \succ c$, then there is some probability p such that the person is indifferent between receiving the intermediate outcome (outcome b) with certainty and a lottery with a probability p of receiving outcome a (the most preferred outcome) and a probability of $(1 - p)$ of receiving the the worst outcome, c.

If we accept the axioms of utility, then there exists a scalar *utility function* which assigns a number on a cardinal scale to each outcome and decision, indicating its relative desirability. Furthermore, it follows that, when there is uncertainty about outcomes, preferred decisions are those that maximize the expected utility, given the probability of alternative outcomes. Thus, the axioms imply the MEU principle.

A.1.5 Decision Analysis

Decision analysis is an engineering discipline that addresses the pragmatics of applying decision theory to real-world problems. The axioms of probability and utility merely dictate a set of formal consistency constraints; they say nothing about how we should elicit or represent a utility function or probability distribution, or about the way we should represent or reason about a decision problem. For example, the axioms have nothing to say about the level of detail we select for representing knowledge in a model, or about the best procedures to enlist to search for and identify a utility-maximizing action.

Decision analysis provides methods that help a decision maker to clarify the problem by explicating decision alternatives, values, and information. In particular, decision scientists have developed a battery of methods for performing *sensitivity analysis*, to help a person identify those uncertainties and assumptions that could have a significant effect on the best action to take, and the expected utility associated with that action. Information-acquisition and reasoning resources, therefore, can be directed to the most important or *sensitive* aspects of the problem.

A.2 Influence Diagrams and Belief Networks

A complete description of a decision problem is called a *decision basis* (Howard and Matheson, 1984). A comprehensive decision basis includes information about alternate actions, the nature and likelihood of possible outcomes, preferences about the outcomes, and relationships among these entities. A variety of representations for a decision basis have been developed in the decision-science community. Many people are familiar with the decision-tree representation of actions, events, and outcomes. Another representation for decision problems is the use of joint probability distributions over variables, in conjunction with a loss function that describes the cost of different actions.

Over the last 10 years, decision analysts and computer scientists, interested in computational applications of decision-theoretic reasoning, have attempted to develop richer knowledge representations that allow people to encode decision bases at different levels of detail. In particular, there has been great interest in allowing people to express qualitative, in addition to quantitative, knowledge about belief, preferences, and decisions. Much of this work has centered on the use of graphical representations called *belief networks* and *influence diagrams.*

The influence diagram is a graphical knowledge-representation language that represents the decision basis (Owen, 1978; Howard and Matheson, 1981; Olmsted, 1983; Shachter, 1986; Shachter, 1988). More specifically, an influence diagram is an acyclic directed graph containing nodes representing propositions (e.g., actions and outcomes), and arcs representing interactions among nodes. Nodes representing propositions are associated with a set of mutually exclusive and exhaustive values that represent alternative possible states. The arcs represent deterministic, probabilistic, and informational relationships between the nodes.

There has been much investigation on a specialization of influence diagrams that do not contain information about decisions and preferences (Rousseau, 1968; Howard and Matheson, 1981; Lemmer, 1983; Cooper, 1984; Pearl and Verma, 1987; Kim and Pearl, 1983). These networks express probabilistic relationships among states of the world exclusively. Several different terms are used for these representations, including *knowledge maps* (Howard, 1989), *causal networks*, *Bayesian nets*, and *belief networks* (Pearl, 1986). We shall use *belief networks.*

A.2.1 Alternate Levels of Representation

The human-oriented expressiveness of influence diagrams and belief networks is based in these representations' three levels of specification: *relation*, *function*, and *number* (Howard and Matheson, 1981). In practice, a person can express relations at one level without explicitly referring to more specific levels. The *relation* level captures the qualitative structure of the problem as expressed in the topology of the network. At this level, the arcs and nodes describe dependencies between the values of propositions or variables (nodes). Each variable in an influence diagram is associated with a set of mutually exclusive and collectively exhaustive values. At the level of *function*, the functional form of the relationships among nodes is specified. At the level of *number*, we specify numerical values that specify details about how the functional

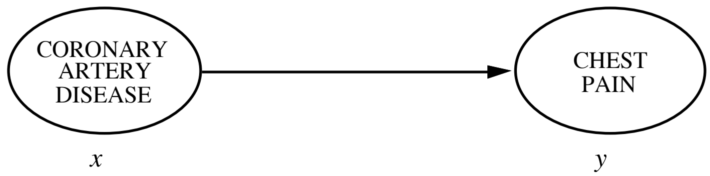


Figure A.1: A simple belief network.
In this belief network, the directed arc specifies a probabilistic dependency between the nodes CORONARY ARTERY DISEASE and CHEST PAIN. Thus, we assert that the probability distribution over values of chest pain is dependent on the severity of coronary artery disease.

forms interact.

Nodes in influence diagrams include chance nodes, decision nodes, and value nodes. Belief networks only contain chance nodes. The chance nodes represent states of the world that are uncertain. We depict chance nodes as circles or ovals. There are two kinds of chance nodes: *stochastic* and *deterministic* (the latter are portrayed as double-lined circles). The belief associated with a stochastic chance node is a probabilistic function of the outcomes of its predecessor nodes. A deterministic chance node is a special case of a stochastic chance node. The values of the predecessors determine the node's value with certainty. Chance nodes without predecessors are specified at the level of number with prior probability distributions.

Figure A.1 displays a simple belief network expressing the uncertain relationship between the degree of coronary artery disease afflicting a patient and that patient's chest pain. The directed arc from the node CORONARY ARTERY DISEASE to the node CHEST PAIN means that the probability distribution over the values of chest pain depends on the value of coronary artery disease.

Figure A.2 shows us more detail about the structure belief network. Associated with each node is a set of mutually exclusive and exhaustive values. The figure also displays the probabilities (as represented by the length of horizontal bars) for each of the four possible values of chest pain, *given* that coronary disease is minor. As shown in Figure A.3, the probability distribution over the different values of chest pain is different when we condition on a more severe form of coronary artery disease.

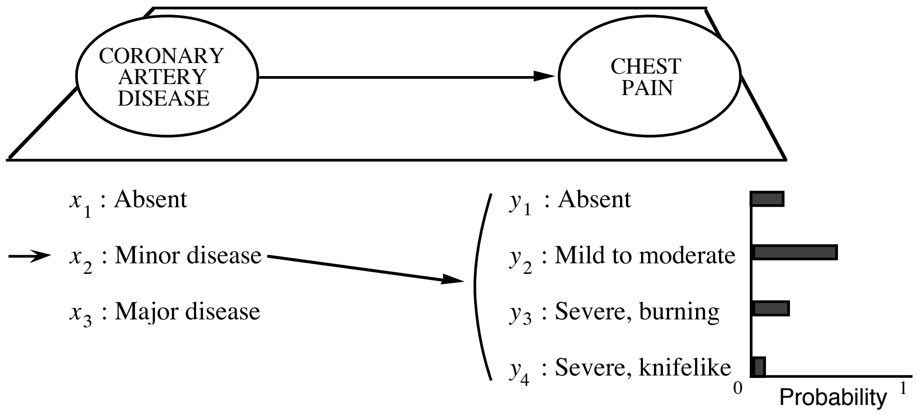


Figure A.2: Values of variables and probability distributions in a belief network. A probability distribution is specified over values of nodes for each state of the predecessor nodes. Here, the figure displays the probability of alternate values of CHEST PAIN conditioned on CORONARY ARTERY DISEASE = Minor disease.

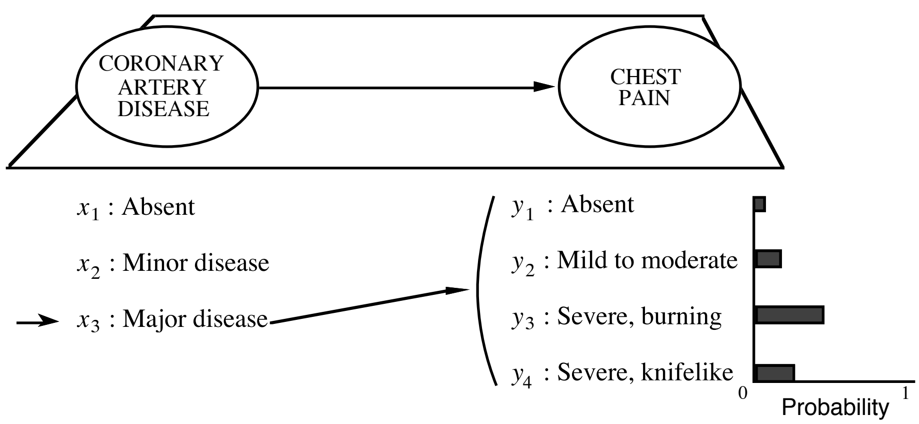


Figure A.3: A probability distribution conditioned on another state. The figure displays the probability of alternate values of CHEST PAIN conditioned on the state CORONARY ARTERY DISEASE = Major disease.

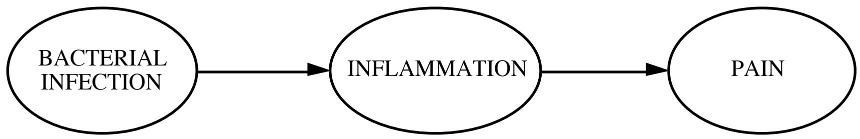


Figure A.4: Conditional independence in a belief network.
In this belief network, assertions about the value of INFLAMMATION renders BACTERIAL INFECTION and PAIN conditionally independent.

A.2.2 Expressing Independence in Belief Networks

Belief networks and influence diagrams are useful for expressing independence among variables. An independence assertion asserts that the belief in proposition Q is not affected by knowledge about the belief in the truth of proposition R, given background information ξ,

$$p(Q|R, \xi) = p(Q|\xi)$$

The lack of arcs among variables—are qualitative expressions of probabilistic independence of various kinds. The independencies in belief networks and influence diagrams are a formal representation of the local nature of the relationships among variables. The graphical representations help us to ascertain the possible effects that one variable can have on a distant variable. In particular, variables with no predecessors or directed pathway between them are marginally independent. Where two variables have one or more common parents, but no arc between them, they are conditionally independent of each other given their common parents. A node is conditionally independent of its indirect predecessors (i.e., nodes at a minimal directed path of distance greater than 1) given all of the variable's immediate predecessors.

In Figure A.4, variables BACTERIAL INFECTION and PAIN are conditionally independent of each other, given an assertion about a value of INFLAMMATION.

A.2.3 Multiple Causation

Let us return to our patient who has been suffering with chest pain. Belief networks are valuable for representing multiple causes among relevant distinctions. For example, an expert physician may tell a decision analyst about several pathophysiological causes of chest pain. In particular, he might point out that, in addition to being

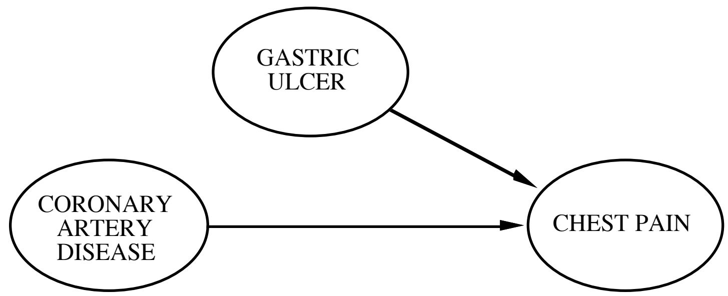


Figure A.5: Representing multiple causes.
We have extended our simple belief network to include knowledge about the relationship between GASTRIC ULCER and CHEST PAIN. Adding this new node and dependency specifies that probability distributions over alternate values of CHEST PAIN are conditioned on states defined by combinations of the values of GASTRIC ULCER and CORONARY ARTERY DISEASE.

caused by coronary artery disease, chest pain also can be caused by a gastric ulcer. In fact, chest pain can be caused by gastric ulcer, by coronary artery disease, or by both ailments acting together. Figure A.5 displays the addition of the new node, GASTRIC ULCER, and a new directed arc, representing the dependency between gastric ulcers and chest pain. Now, the assessments for chest pain must be conditioned simultaneously on *combinations* of all of the possible values of CORONARY ARTERY DISEASE and GASTRIC ULCER.

A.2.4 Actions and Preferences

A physician examining a patient, with a chief complaint of chest pain, might desire to test the patient to learn more about the likelihoods of coronary artery disease and chest pain. On the other hand, the physician may recommend that the patient take immediate action to treat one or both of the possible causes of chest pain. Testing or treatment decisions are captured in the richer representation of influence-diagrams. Figure A.6 displays an extension of the belief network in Figure A.5 to a simple influence diagram. Note that we have added a decision node (square node), representing alternative decisions, and a value node (diamond) capturing information about the preferences of the patient.

The arc pointing into the decision node in Figure A.6 is called an *informational*

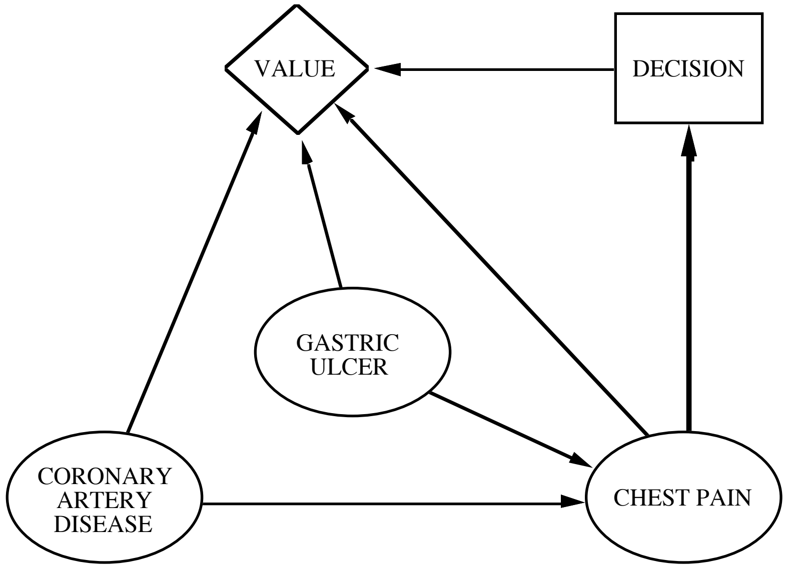


Figure A.6: An influence diagram for treating chest pain.
This influence diagram represents the decision basis for a decision about the treatment of chest pain that may be caused by GASTRIC ULCER, CORONARY ARTERY DISEASE, or by both ailments. In addition to the chance nodes (ovals), we introduce a decision node (square), and a value node (diamond). We also add an informational arc (thicker), representing information that is known at the time a decision is made.

arc. Informational arcs represent knowledge about the world that is known at the time a decision is made. In this case, the physician has information from the patient about the nature of his chest pain. Notice that the value of alternative treatment and testing actions is a function of the decision and the values of coronary artery disease and gastric ulcer.

A decision analyst attempts to carefully refine a decision model to ensure that all important factors are taken into consideration. The process of model building often involves an iterative cycle consisting of the repeated modification or pruning away of previously identified nodes and the addition of new nodes and arcs. Figure A.7 represents a richer influence diagram that might be constructed to decide if a patient should undergo surgery to repair his potential coronary artery disease. This influence diagram includes distinctions about the probabilistic relationship of coronary artery disease to myocardial infarction (heart attack). Also, important

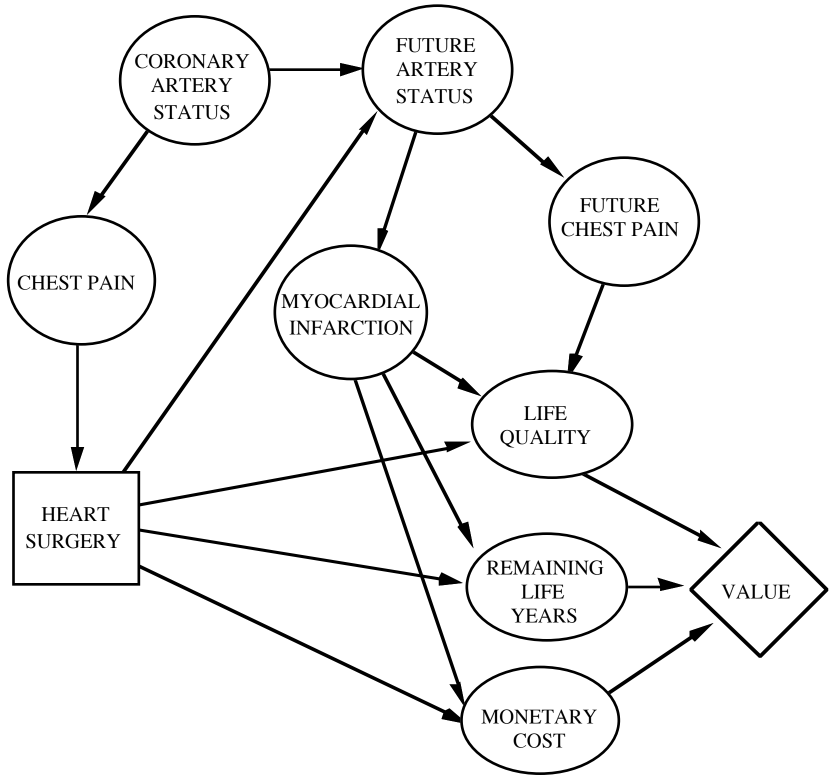


Figure A.7: An influence diagram for a decision about heart surgery.
This influence diagram represents a decision basis for analyzing the value of surgery that would attempt to increase the flow of blood to the heart. Components of value explicitly represented in the diagram include LIFE QUALITY, REMAINING LIFE YEARS, and MONETARY COST.

components of the utility model are explicitly broken out to facilitate an analysis of the patient's preferences. In this case, relevant attributes include life quality, life years, and the monetary cost of the angiogram test, the cost of surgery, and the cost of hospitalization if a myocardial infarction occurs. The diagram shows that the quality of life is influenced by the level of chest pain and the morbidity of the surgery. The value function is a real-valued scalar function that represents tradeoffs among alternative attributes for a patient, as well as individual preferences about risk and time.

It is frequently useful to gather additional information about important variables before taking actions that can have dramatic effects on a patient's utility. In the

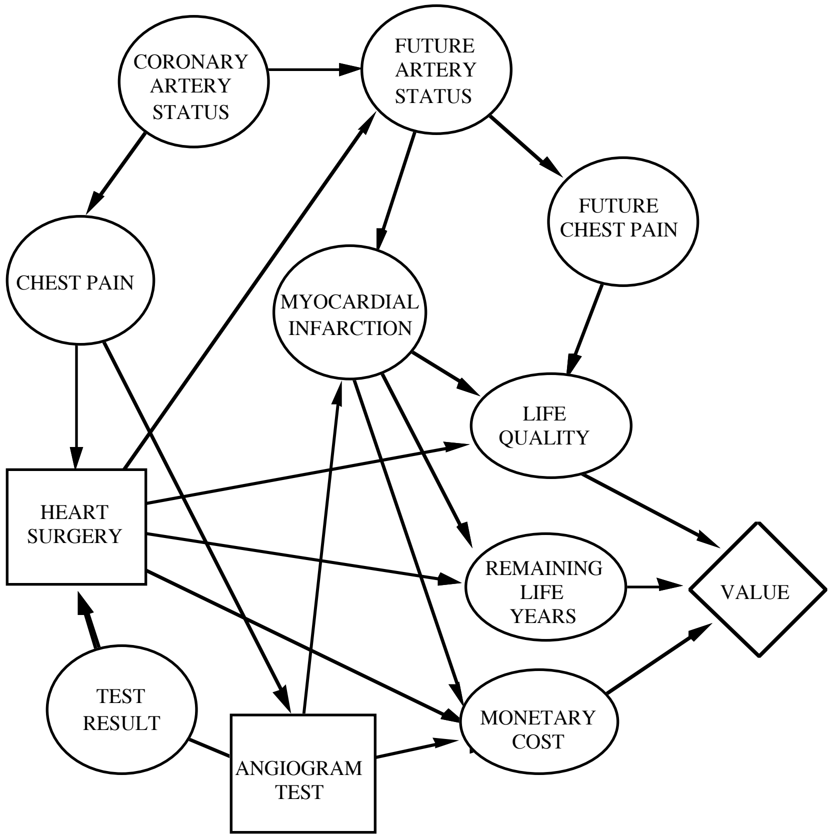


Figure A.8: Reasoning about the value of gathering additional information. This influence diagram considers the value, monetary cost, and risk associated with performing a potentially dangerous imaging test to gather additional information about CORONARY ARTERY STATUS. The decision to receive heart surgery could be sensitive to the result of such a test.

case of an elderly patient complaining of chest pain, we might wish to perform a test to gather more information about the likelihood of alternate values of coronary artery disease. Figure A.8 shows an extension of the influence diagram, displayed in Figure A.7, that captures the preliminary decision to perform a potentially dangerous angiogram test to image the coronary arteries. If we do perform the test, we shall know the result before we make a decision about whether to perform surgery.

A.3 Inference Algorithms

Over the last 5 years, there has been substantial progress on algorithms for solving belief networks and influence diagrams. Belief-network algorithms allow us to draw conclusions about variables of interest in the networks, given information or assumptions about the value of other variables. A common application of belief network algorithms is to compute the marginal probability distribution for a proposition, given the value of other variables. For example, we may wish to determine the probability of myocardial infarction for a specific patient, given information about his medical history, and observations about his current pain and physical endurance. There are several classes of algorithm for probabilistic inference in belief networks. Alternatively, we may wish to determine directly the best decision to make, given the current state of information. Investigators have developed algorithms that work directly on influence diagrams to address this task.

A.3.1 Exact methods

Exact methods are algorithms that precisely solve belief-network problems, yielding point probabilities of interest. Cooper performed a complexity analysis of probabilistic inference, and demonstrated that the problem of probabilistic inference in belief networks is $\mathcal{NP}$-hard (Cooper, 1990b). This result highlights the futility of developing an exact method that is computationally efficient for arbitrary belief networks. Nevertheless, progress has been made on exact methods for the tractable solution of specific belief-network topologies. Most of these methods take advantage of the special structure and sparseness in the interconnectivity of sample networks.

A belief network with probabilities assigned to all nodes and influences specifies a complete joint probability distribution over the variables in the network. We can generate the joint distribution defined by a belief network in a brute-force manner by taking the product of all of the assigned distributions. Once we generate the joint distribution, we can compute the marginal and conditional probabilities of interest by summing over the relevant dimensions of the joint distribution. Although the brute-force approach is conceptually straightforward, it requires computational effort that is exponential in the number of variables.

The key to efficient exact computation in belief networks is to exploit independencies to avoid having to calculate the full joint probability distribution. Kim and Pearl have developed a distributed algorithm for solving singly connected networks, or *polytrees* (Kim and Pearl, 1983). The algorithm is linear in the number of variables in the network. In the distributed approach, each node in the network receives messages from each of its parent and child nodes, representing all the evidence available from alternative portions of the network. Each time a new observation is made, messages are propagated through the network to update the probabilities of the values of other variables.

Another approach, developed by Pearl, called the method of conditioning relies on the transformation of multiply connected networks to a set of singly connected networks (Pearl, 1986) (this method is described in more detail in Chapter 5). The method relies on the identification of a loop cutset, that "breaks" all of the cycles in a network. Loop-cutset nodes must be instantiated with each possible value (or combination of values). Once the method of conditioning has been applied, techniques for solving singly connected networks (such as the Kim and Pearl distributed polytree algorithm) can be applied to solve the polytree network subproblems.

An arc-reversal method developed by Shachter (Shachter, 1988) applies a sequence of operators to reverse the arcs in a belief network. Arc reversal is equivalent to applying Bayes' rule. Through a series of arc reversals and node removal, a belief network can be reduced to one node, representing the answer to a probabilistic query. Shachter's algorithm can be applied to multiply connected networks, but requires detailed knowledge about topology.

Lauritzen and Spiegelhalter have developed an approach based on the reformulation of a belief network into a tree by forming clusters of nodes or cliques (Lauritzen and Spiegelhalter, 1987; Lauritzen and Spiegelhalter, 1988). A clique is defined as a set of nodes where each node in the set has an arc to all other nodes in the set. Any network can be converted into a corresponding singly connected network of cliques. The Lauritzen and Spiegelhalter approach makes use of an algorithm for propagating evidence within a tree of cliques. This approach is linear in the number of cliques and exponential in the size of the largest clique. Although this algorithm is one of the fastest available, it quickly becomes intractable as the connectedness of a network increases.

In other work, Heckerman has developed an algorithm named Quickscore for performing inference in a special two-level diagnostic network consisting of binary variables (Heckerman, 1989). The algorithm relies on an assumption of causal independence (the noisy-or assumption (Pearl, 1988)) in the special diagnostic model. Quickscore has a complexity that is exponential in the number of positive findings (findings observed to be present).

Recently, Cooper developed an approach to exact inference called the *method of dissection* (Cooper, 1990a). His approach relies on a method for recursively decomposing a belief network into a binary tree. Then, inference is performed over the tree. Cooper describes a heuristic procedure that searches among alternative decompositions to find a decomposition associated with the smallest number of arithmetic operations to calculate a probability of interest.

A.3.2 Approximation Methods

All of the exact methods are highly sensitive to the connectedness of belief networks. Inference approximation methods promise to solve networks more quickly. However, the performance of these methods often escape crisp a priori characterization. Approximation methods include stochastic simulation and probability-bounding algorithms.

A.3.2.1 Stochastic Simulation

Simulation techniques estimate the probability of an event based on the frequency with which that event occurs in a set of simulation trials. Some simulation methods report a probability distribution or partial characterization of a distribution over probabilities of interest.

Henrion introduced the notion of using simulation to solve belief-network inference with a technique called logic sampling (Henrion, 1988). Convergence rates of logic sampling degrade exponentially with the number of pieces of evidence considered in a problem. Several promising simulation algorithms have been developed recently, including Chavez and Cooper's randomized polynomial method (Chavez and Cooper, 1989a) and the likelihood-weighting approaches of Shachter and Peot (Shachter and Peot, 1989), and of Fung and Chang (Fung and Chang, 1989). The likelihood-weighting approaches employ techniques for directing the simulation to generate and analyze cases that are more likely to explain evidence.

A.3.2.2 Bounding

There has been ongoing interest in the calculation of upper and lower *bounds* on point probabilities of interest. Probabilistic bounding techniques determine bounds on probabilities through a logical analysis of constraints acquired during a partial analysis. Bounds become tighter as additional constraints are brought into consideration. Cooper (Cooper, 1984) and Peng (Peng, 1986) have investigated the use of a best-first search algorithm to focus attention on the most relevant aspects of the problem in calculating bounds on the hypotheses. These methods are able to prune the search dramatically by eliminating all extensions of a diagnosis that are provably less probable than the current best.

In more recent work, Henrion developed a bounding method named *TopN* that relies on search techniques in a special two-level network (Henrion, 1990). A version of the algorithm can be used to identify the most probable hypotheses. Work is in progress on the characterization of this method.

In Chapter 5, I describe a bounding method algorithm named *bounded conditioning* (Horvitz et al., 1989c). Bounded conditioning is based on Pearl's method of conditioning, described in Section A.3.1. With this method, bounds are generated on probabilities of interest through modulating the completeness of an analysis. In particular, we vary the fraction of polytree subproblems that are solved, and bound the possible contributions of the unsolved subproblems.

A.3.3 Algorithms for Inference in Influence Diagrams

A straightforward method to solve a well-structured influence diagram is to convert the influence diagram into a corresponding decision tree, and to solve that tree. We can solve a decision tree by computing the utility for each terminal node. The well-known "roll-back" method for solving decision trees computes the expected utility over the branches at each outcome variable, and the maximum expected utility over the alternatives at each decision. In the worst case, the tree-processing algorithm is exponential in the number of outcome and decision variables.

Shachter has developed a method that operates directly on influence diagrams (Shachter, 1986). The algorithm applies a sequence of operations to the diagram, successively eliminating nodes when their effects have been accounted for through expected value calculations. The operations correspond to applying Bayes' theorem (corresponding to an arc-reversal operation), forming conditional expectations (equivalent to removing a chance node), and maximizing the expected utility (equivalent to removing a decision node). The algorithm's identifies the optimal decisions, conditioned on the information available when each decision is made, and the expected value of the decision strategies. Unfortunately, the algorithm is liable to the same $\mathcal{NP}$-hardness problem that hinders exact probabilistic inference.

Appendix B

Glossary of Belief-Network Abbreviations

A VENT: Alveolar ventilation

A VENT/BREATH: Alveolar ventilation per breath

BP: Blood pressure

CHF: Congestive heart failure

CO: Cardiac output

CO2 PRODUCTION: Rate of production of carbon dioxide by the tissues

CVP: Central venous pressure

DISCONNECTION: Indication if the ventilator is disconnected from the patient

ERROR CAUTER: Error in HR from EKG because of cautery use

ERROR LOW OUTPUT: Error in HR from the blood pressure monitor or from the pulse oximeter due to a low cardiac output

FIO2: Fraction of inspired oxygen

HGB: Hemoglobin (also referred to as Hb in medicine)

HR: Heart rate

HR BP: Heart rate from the arterial blood-pressure measuring device

HR EKG: Heart rate determined from the electrocardiograph (EKG)

HR SAT: Heart rate from the pulse oximeter

LVED VOLUME: Left-ventricular end-diastolic volume

MAP: Mean arterial pressure

MV: Minute ventilation

MV SETTING: Minute ventilation set at the ventilator as the product of the set tidal volume and respiratory rate

O2 CONCENTRATION: Fraction of inspired oxygen (also called FIO2)

O2 CONSUMPTION: Rate of oxygen consumed by the tissues

O2 DELIVERY: Oxygen delivered to the tissues and available for consumption

PACO2: Partial pressure of carbon dioxide in the arterial blood

PAO2: Partial pressure of oxygen in the arterial blood

PAP: Mean pulmonary artery pressure

PAPO2: Partial pressure of oxygen in the arterial blood

PA SAT: Pulmonary venous (highly oxygenated) oxygen saturation.

PCWP: Pulmonary capillary wedge pressure (also called pulmonary artery wedge pressure [PAW])

PEEP: Positive end expiratory pressure

PRESSURE: Peak inspirational pressure delivered by the ventilator

PVCO2: Pulmonary venous carbon-dioxide tension

PVO2: Pulmonary venous oxygen tension

RQ: Respiratory quotient, the ratio of oxygen consumption to carbon dioxide production

SAO2: Level of oxygen saturation in the arterial blood

TEMP: Body temperature

TPR: Total peripheral resistance (also called systemic vascular resistance [SVR])

VENT ALV: Ventilation at the alveoli

VENT LUNG: Ventilation at the major airways of the lung (bronchi)

VENT MACHINE: Ventilation at the ventilation machine

VENT TUBE: Ventilation at the ventilation tube

Bibliography

Aggarwal, A. and Vitter, J. (1988). The input/output complexity of sorting and related problems. *CACM*, 31(9):1116–1127.

Agogino, A. and Ramamurthti, K. (1989). Real-time reasoning about time constraints and model precision in complex distributed mechanical systems. In *Working Notes: Spring Symposium Series on AI and Limited Rationality*, pages 1–5, Stanford University. American Association for Artificial Intelligence.

Agre, P. and Chapman, D. (1987). Pengi: An implementation of a theory of activity. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, pages 268–272, Seattle, WA. AAAI-87.

Aho, A., Hopcroft, J., and Ullman, J. (1983). *Data Structures and Algorithms*. Addison-Wesley, Menlo Park, California.

Andreassen, S., Woldbye, M., Falck, B., and Andersen, S. (1987). MUNIN—a causal probabilistic network for interpretation of electromyographic findings. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence, Milan, Italy*, pages 366–372. Morgan Kaufman, San Mateo, CA.

Barnett, J. (1984). How much is control knowledge worth? *Journal of Artificial Intelligence*, 22:77–89.

Barry, M., Mulley, Jr., A. Fowler, F., and Wennberg, J. (1988). Watchful waiting versus immediate transurethral resection for symptomatic prostatism: The importance of patients' preferences. *Journal of the American Medical Association*, 259:3010–17.

Bayes, T. (1958). An essay towards solving a problem in the doctrine of chances. *Biometrika*, 46:293–298. Reprint of Bayes' 1763 manuscript.

Beinlich, I., Suermondt, H., Chavez, R., and Cooper, G. (1989). The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the Second European Conference on Artificial Intelligence in Medicine, London*. Springer Verlag, Berlin.

Ben-Bassat, M. (1978). Myopic policies in sequential classification. *IEEE Transactions on Computers*, 27:170–178.

Ben-Bassat, M. and Teeni, D. (1984). Human-oriented information acquistion in sequential pattern classification: Part 1 - single membership classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 14:131–138.

Boddy, M. and Dean, T. (1989). Solving time-dependent planning problems. In *Proceedings of the Eleventh IJCAI*. AAAI/International Joint Conferences on Artificial Intelligence.

Breese, J. (1987). *Knowledge Representation and Inference in Intelligent Decision Systems*. PhD thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.

Breese, J. (1990). Construction of belief and decision networks. Technical Report Technical Memorandum 30, Rockwell International Science Center, Palo Alto, California.

Breese, J. and Horvitz, E. (1990). Ideal reformulation of belief networks. In *Proceedings of Sixth Conference on Uncertainty in Artificial Intelligence, Cambridge, MA*, pages 64–72. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Brooks, R. (1987). Planning Is Just a Way of Avoiding Figuring Out What to Do Next. Technical Report Working Paper 303, M.I.T. Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Bruner, J., Goodnow, J., and Austin, G. (1956). *A study of thinking*. Wiley and Sons.

Buchanan, B. (1966). *Logics of Scientific Discovery*. PhD thesis, University of Michigan.

Buchanan, B. and Shortliffe, E., editors (1984). *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA.

Buchanan, B. G. (1982). Research on expert systems. In J. Hayes, D. Michie, Y. H. P., editor, *Machine Intelligence*, volume 10, pages 269–299. Ellis Howard Ltd., Chichester, England.

Charniak, E. and Goldman, R. (1989). A semantics for probabilistic quantifier-free first-order languages, with particular application to story understanding. In *Proceedings Eleventh IJCAI*, Detroit, Michigan. International Joint Conferences on Artificial Intelligence.

Chavez, R. and Cooper, G. (1989a). An empirical evaluation of a randomized algorithm for probabilistic inference. In *Proceedings of Fifth Workshop on Uncertainty in Artificial Intelligence, Windsor, ON*, pages 60–70. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Chavez, R. and Cooper, G. (1989b). A randomized approximation algorithm for probabilistic inference on Bayesian belief networks. Technical Report KSL-88-72, Medical Computer Science Group, Section on Medical Informatics, Stanford University, Stanford, CA.

Clancey, W. (1985). Heuristic classification. *Artificial Intelligence*, 27:289–350.

Cohen, P. R., DeLisio, J. L., and Hart, D. M. (1989). A declarative representation of control knowledge. *IEEE Trans. on Systems, Man and Cybernetics*, 19(3):546–557.

Coles, L., Robb, A., Sinclair, P., Smith, M., and Sobek, R. (1975). Decision analysis for an experimental robot with unreliable sensors. In *Proceedings of the Fourth International Joint Conference on Artificial Intelligence, Georgia, USSR*, pages 749–757. International Joint Conference on Artificial Intelligence.

Cooper, G. (1984). *NESTOR: A Computer-based Medical Diagnostic Aid that Integrates Causal and Probabilistic Knowledge.* PhD thesis, Computer Science Department, Stanford University, Stanford, CA. Rep. No. STAN-CS-84-48. Also numbered HPP-84-48.

Cooper, G. (1990a). Bayesian belief-network inference using nested dissection. Technical Report KSL-90-05, Stanford University.

Cooper, G. (1990b). Probabilistic inference using belief networks is NP-hard. *Artificial Intelligence*, 42:393–405.

Cox, R. (1946). Probability, frequency and reasonable expectation. *American Journal of Physics*, 14:1–13.

David, F. (1962). *Games, Gods, and Gambling.* Charks Griffen and Company, London.

Davis, R. (1982). Consultation, knowledge acquisition, and instruction. In Szolovits, P., editor, *Artificial Intelligence In Medicine*, pages 57–78. Westview Press, Boulder, CO.

Dawes, R. and Corrigan, B. (1974). Linear models in decision making. *Psychological Bulletin*, 81:95–106.

de Dombal, F., Leaper, D., Horrocks, J., Staniland, J., and McCain, A. (1974). Human and computer-aided diagnosis of abdominal pain: further report with emphasis on performance. *British Medical Journal*, 1:376–380.

de Dombal, F., Leaper, D., Staniland, J., McCann, A., and Horrocks, J. (1972). Computer-aided diagnosis of acute abdominal pain. *British Medical Journal*, 2:9–13.

de Finetti, B. (1970). *Theory of Probability.* Wiley and Sons, New York.

Dean, T. and Boddy, M. (1988). An analysis of time-dependent planning. In *Proceedings AAAI-88 Seventh National Conference on Artificial Intelligence*, pages 49–54. American Association for Artificial Intelligence.

Dean, T. and Kanazawa, K. (May, 1988). Probabilistic temporal reasoning. Technical report, Brown University.

Doyle, J. (1988). Artificial intelligence and rational self-government. Technical Report CS-88-124, Carnegie-Mellon University.

Durfee, E. and Lesser, V. (1987). Planning to meet deadlines in a blackboard-based problem solver. In Stankovic, J. . K. R., editor, *Tutorial on Hard Real-Time Systems*, pages 595–608. IEEE Computer Society Press.

Elstein, A., Loupe, M., and Erdman, J. (1971). An experimental study of medical diagnostic thinking. *Journal of Structural Learning*, 2:45–53.

Elstein, A., Shulman, L., and Sprafka, S. (1978). *Medical Problem Solving: An Analysis of Clinical Reasoning.* Harvard University Press, Cambridge, MA.

Erman, L., Hayes-Roth, F., Lesser, V., and Reddy, D. (1980). The HEARSAY-II speech understanding system: integrating knowledge to resolve uncertainty. *ACM Computing Surveys*, 12:213–253.

Etzioni, O. and Mitchell, T. (1989). A comparative analysis of chunking and decision-analytic control. In *Working Notes: Spring Symposium Series on AI and Limited Rationality*, pages 42–45, Stanford University. American Association for Artificial Intelligence.

Fehling, M. and Breese, J. (1988). A computational model for the decision-theoretic control of problem solving under uncertainty. Technical Report Rockwell Technical Report 837-88-5, Rockwell International Science Center.

Feigenbaum, E. (1964). *Computers and Thought*. McGraw-Hill, New York.

Feldman, J. R. and Sproull, R. F. (1975). Decision theory and artificial intelligence ii: The hungry monkey. *Cognitive Science*, 1:158–192.

Fishburn, P. (1981). Subjective expected utility: A review of normative theories. *Theory and Decision*, 13:139–199.

Fung, R. and Chang, K. (1989). Weighting and integrating evidence for stochastic simulation in Bayesian networks. In *Proceedings of Fifth Workshop on Uncertainty in Artificial Intelligence, Windsor, ON*, pages 112–117. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Gaines, B. R. (1978). Fuzzy and probability uncertainty logics. *Information and Control*, 38:154–169.

Garey, M. and Johnson, D. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York.

Ginsberg, M. (1987). *Readings in Nonmonotonic Logic*. Morgan Kaufman, San Mateo, CA.

Good, I. (1950). *Probability and the Weighing of Evidence*. Hafners, New York.

Good, I. (1952). Rational decisions. *J. R. Statist. Soc. B*, 14:107–114.

Good, I. (1977). Dynamic probability, computer chess, and the measurement of knowledge. In Elcock, E. and D., M., editors, *Machine Intelligence*, pages 139–150. Wiley, New York.

Gorry, G. (1973). Computer-assisted clinical decision making. *Methods of Information in Medicine*, 12:45–51.

Gorry, G. and Barnett, G. (1968). Experience with a model of sequential diagnosis. *Computers and Biomedical Research*, 1:490–507.

Hacking, I. (1975). *The Emergence of Probability*. Cambridge University Press, Cambridge.

Hansson, O. and Mayer, A. (1989). Probabilistic heuristic estimates. In *Proceedings of the Second International Workshop on AI and Statistics*, Ft. Lauderdale, FL.

Hayes-Roth, B. (1985). A blackboard architecture for control. *Artificial Intelligence*, 26(2):251–321.

Heckerman, D. (1986). Probabilistic interpretations for MYCIN's certainty factors. In Kanal, L. and Lemmer, J., editors, *Uncertainty in Artificial Intelligence*, pages 167–196. North Holland, New York.

Heckerman, D. (1989). A tractable algorithm for diagnosing multiple diseases. In *Proceedings of Fifth Workshop on Uncertainty in Artificial Intelligence, Windsor, ON*, pages 174–181. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Heckerman, D. (1990a). An empirical comparison of three inference methods. In Levitt, R. S. T., Lemmer, J., and Kanal, L., editors, *Uncertainty in Artificial Intelligence 4*. North Holland, New York.

Heckerman, D. (1990b). *Probabilistic Similarity Networks*. PhD thesis, Medical Computer Science Group, Section on Medical Informatics, Stanford University, Stanford, CA.

Heckerman, D., Breese, J., and Horvitz, E. (1989a). The compilation of decision models. In *Proceedings of Fifth Workshop on Uncertainty in Artificial Intelligence, Windsor, ON*, pages 162–173. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Heckerman, D. and Horvitz, E. (1987). On the expressiveness of rule-based systems for reasoning under uncertainty. In *Proceedings AAAI-87 Sixth National Conference* *on Artificial Intelligence, Seattle, WA*, pages 121–126. Morgan Kaufmann, San Mateo, CA.

Heckerman, D. and Horvitz, E. (1990). Problem formulation as the reduction of a decision problem. In *Proceedings of Sixth Conference on Uncertainty in Artificial Intelligence, Cambridge, MA*. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Heckerman, D., Horvitz, E., and Nathwani, B. (1985). Pathfinder research directions. Technical Report KSL-89-64, Medical Computer Science Group, Section on Medical Informatics, Stanford University, Stanford, CA.

Heckerman, D., Horvitz, E., and Nathwani, B. (1989b). Update on the Pathfinder project. In *Proceedings of the Thirteenth Symposium on Computer Applications in Medical Care, Washington, DC*, pages 203–207. IEEE Computer Society Press, Los Angeles, CA.

Heckerman, D., Horvitz, E., and Nathwani, B. (1990). Toward normative expert systems: The Pathfinder project. Technical Report KSL-90-08, Medical Computer Science Group, Section on Medical Informatics, Stanford University, Stanford, CA. Submitted to *Artificial Intelligence in Medicine*.

Heckerman, D. and Jimison, H. (1989). A perspective on confidence and its use in focusing attention during knowledge acquisition. In Kanal, L., Levitt, T., and Lemmer, J., editors, *Uncertainty in Artificial Intelligence 3*, pages 123–131. North Holland, New York.

Henrion, M. (1988). Propagation of uncertainty by probabilistic logic sampling in Bayes' networks. In Lemmer, J. and Kanal, L., editors, *Uncertainty in Artificial Intelligence 2*, pages 149–164. North Holland, New York.

Henrion, M. (1990). Towards efficient probabilistic diagnosis in multiply connected networks. In Oliver, R. and Smith, J., editors, *Influence Diagrams, Belief Networks, and Decision Analysis*, pages 385–409. John Wiley and Sons, Chichester.

Herskovits, E. and Cooper, G. (1989). Algorithms for belief-network precomputation. Technical Report KSL-89-35, Stanford University.

Holtzman, S. (1985). *Intelligent Decision Systems.* PhD thesis, Department of Engineering–Economic Systems, Stanford University, Stanford, CA.

Holtzman, S. (1989). *Intelligent Decision Systems.* Addison–Wesley, Menlo Park, CA.

Horvitz, E. (1986). Toward a science of expert systems. *Proceedings of the 18th Symposium on the Interface of Computer Science and Statistics*, pages 45–52.

Horvitz, E. (1987a). A multiattribute utility approach to inference understandability and explanation. Technical Report KSL-28-87, Medical Computer Science Group, Section on Medical Informatics, Stanford University, Stanford, CA.

Horvitz, E. (1987b). Problem-solving design: Reasoning about computational value, tradeoffs, and resources. In *Proceedings of the NASA Artificial Intelligence Forum*, pages 26–43, Palo Alto, CA.

Horvitz, E. (1987c). Reasoning about beliefs and actions under computational resource constraints. In *Proceedings of Third Workshop on Uncertainty in Artificial Intelligence*, Seattle, Washington. American Association for Artificial Intelligence. Also in L. Kanal, T. Levitt, and J. Lemmer, ed., *Uncertainty in Artificial Intelligence 3,* Elsevier, 1989, pps. 301-324.

Horvitz, E. (1988). Reasoning under varying and uncertain resource constraints. In *Proceedings AAAI-88 Seventh National Conference on Artificial Intelligence, Minneapolis, MN*, pages 111–116. Morgan Kaufmann, San Mateo, CA.

Horvitz, E. (1989a). Rational metareasoning and compilation for optimizing decisions under bounded resources. In *Proceedings of Computational Intelligence 89.* Association for Computing Machinery. Also available as Technical Report KSL-89-81 Knowledge Systems Laboratory, Stanford University, April 1989.

Horvitz, E. (1989b). Reasoning about beliefs and actions under computational resource constraints. In Kanal, L., Levitt, T., and Lemmer, J., editors, *Uncertainty in Artificial Intelligence 3*, pages 301–324. North Holland, New York. Also in *Proceedings of Third Workshop on Uncertainty in Artificial Intelligence*, Seatle, Washington, 1987.

Horvitz, E. and Breese, J. (1990). Ideal partition of resources for metareasoning. Technical report, Knowledge Systems Laboratory, Stanford University. KSL-90-26.

Horvitz, E., Cooper, G., and Heckerman, D. (1989a). Reflection and action under scarce resources: Theoretical principles and empirical study. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, Detroit, MI*, pages 1121–1127. International Joint Conference on Artificial Intelligence.

Horvitz, E., Heckerman, D., and Langlotz, C. (1986a). A framework for comparing alternative formalisms for plausible reasoning. In *Proceedings AAAI-86 Fifth National Conference on Artificial Intelligence, Philadelphia, PA*, pages 210–214. Morgan Kaufmann, San Mateo, CA.

Horvitz, E., Heckerman, D., Nathwani, B., and Fagan, L. (1986b). The use of a heuristic problem-solving hierarchy to facilitate the explanation of hypothesis-directed reasoning. In *Proceedings of Medinfo, Washington, DC*, pages 27–31. North Holland, New York.

Horvitz, E., Heckerman, D., Ng, K., and Nathwani, B. (1989b). Heuristic abstraction in the decision-theoretic Pathfinder system. In *Proceedings of the Thirteenth Symposium on Computer Applications in Medical Care, Washington, DC*, pages 178–182. IEEE Computer Society Press, Los Angeles, CA.

Horvitz, E., Suermondt, H., and Cooper, G. (1989c). Bounded conditioning: Flexible inference for decisions under scarce resources. In *Proceedings of Fifth Workshop on Uncertainty in Artificial Intelligence, Windsor, ON*, pages 182–193. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Howard, R. (1966). Decision analysis: Applied decision theory. In Hertz, D. and Melese, J., editors, *Proceedings of the Fourth International Conference on Operational Research*, pages 55–71. Wiley-Interscience.

Howard, R. (1970). Decision analysis: Perspectives on inference, decision, and experimentation. *Proceedings of the IEEE*, 58:632–643.

Howard, R. (1980). On making life and death decisions. In Howard, R. and Matheson, J., editors, *Readings on the Principles and Applications of Decision Analysis*, volume II, pages 483–506. Strategic Decisions Group, Menlo Park, CA.

Howard, R. (1988). Decision analysis: Practice and promise. *Management Science*, 34:679–695.

Howard, R. (1989). Knowledge maps. *Management Science*, 35(8):903–922.

Howard, R. and Matheson, J. (1981). Influence diagrams. In Howard, R. and Matheson, J., editors, *Readings on the Principles and Applications of Decision Analysis*, volume II, pages 721–762. Strategic Decisions Group, Menlo Park, CA.

Howard, R. and Matheson, J., editors (1984). *Readings on the Principles and Applications of Decision Analysis*. Strategic Decisions Group, Menlo Park, Ca.

Jacobs, W. and Keifer, M. (1973). Robot decisions based on maximizing utility. In *Proceedings of the Third International Joint Conference on Artificial Intelligence, ??*, pages 402–411. International Joint Conference on Artificial Intelligence.

Jimison, H. (1990). Generating explanations of decision models based on an augmented representation of uncertainty. In Shachter, R., Kanal, L., Levitt, T., and Lemmer, J., editors, *Uncertainty in Artificial Intelligence 4*. North Holland, New York. in press.

Kaelbling, L. (1987). An architecture for intelligent reactive systems. In Lansky, A. and Georgeff, M., editors, *Reasoning About Actions and Plans: Proceedings of the 1986 Workshop*, pages 395–410. Morgan-Kaufmann.

Kahneman, D., Slovic, P., and Tversky, A., editors (1982). *Judgment Under Uncertainty: Heuristics and Biases.* Cambridge University Press, New York.

Kanazawa, K. and Dean, T. (1989). A model for projection and action. In *Proceedings of the Eleventh IJCAI.* AAAI/International Joint Conferences on Artificial Intelligence.

Keeney, R. and Raiffa, H. (1976). *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* Wiley and Sons, New York.

Kim, H., Zelman, R., Fox, M., Bennett, J., Berard, C., Butler, J., Byrne, G., Dorfman, R., Hartsock, R., Lukes, R., Mann, R., Neiman, R., Rebuck, J., Sheehan, W., Variakojis, D., Wilson, J., and Rappaport, H. (1982). Pathology panel for lymophoma clinical studies: A comprehensive analysis of cases accumulated since its inception. *Journal of the National Cancer Institute*, 68:43–67.

Kim, J. and Pearl, J. (1983). A computational model for causal and diagnostic reasoning in inference engines. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence, Karlsruhe, West Germany*, pages 190–193. International Joint Conference on Artificial Intelligence.

Klein, D. (1987). Explaining and refining decision-theoretic choices. Technical Report MS-CIS-87-57 (LINC 74), Dept. of Computer and Information Science, University of Pennsylvania.

Klein, D. (1989). *Explaining and Refining Decision-Theoretic Choices.* PhD thesis, Dept. of Computer and Information Science, University of Pennsylvania.

Knuth, D. (1973). *Sorting and Searching.* Addision-Wesley, Reading, Massachusetts.

Kripke, S. (1975). Outline of a theory of truth. *Journal of Philosophy*, 72:690–716.

Laffey, T., Cox, P., Schmidt, J., Kao, S., and Read, J. (1988). Real-time knowledge-based systems. *AI Magazine*, Spring:27ff.

Langlotz, C., Shortliffe, E., and Fagan, L. (1986). A methodology for computer-based explanation of decision analysis. Technical Report KSL-86-57, Stanford University.

Langlotz, C. P., Shortliffe, E. H., and Fagan, L. M. (1988). A methodology for generating computer-based explanations of decision-theoretic advice. *Medical Decision Making*, 8(4):290–303.

Lauritzen, S. and Spiegelhalter, D. (1987). Fast manipulation of probabilities with local representations with applications to expert systems. Technical Report R-87-7, Institute of Electronic Systems, Aalborg University, Aalborg, Denmark.

Lauritzen, S. and Spiegelhalter, D. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *J. Royal Statistical Society B*, 50:157–224.

Lawler, E., Lenstra, J., Kan, A. R., and Shmoys, D. (1985). *The Traveling Salesman Problem*. John Wiley and Sons, New York.

Ledley, R. and Lusted, L. (1959). Reasoning foundations of medical diagnosis. *Science*, 130:9–21.

Lehman, R. (1955). On confirmation and rational betting. *Journal of Symbolic Logic*, 20:251–262.

Lemmer, J. (1983). Generalized Bayesian updating of incompletely specified distributions. *Large Scale Systems*, 5.

Lindberg, D., Sharp, G., Kingsland, L., Weiss, S., Hayes, S., Ueno, H., and Hazelwood, S. (1980). Computer-based rheumatology consultant. In *Proceedings of Medinfo*, pages 1311–1315. North Holland, New York.

Lindley, D. (1982). Scoring rules and the inevitability of probability. *International Statistical Review*, 50:1–26.

Lipman, B. (1989). How to decide how to decide hot to...: Limited rationality in decisions and games. Technical report, Carnegie Mellon University, Pittsburgh.

Logan (1985). *The Value of Probability Assessment*. PhD thesis, Department of Engineering-Economic Systems, Stanford University.

March, J. (1978). Bounded rationality, ambiguity, and the engineering of choice. *Bell Journal of Economics*, pages 587–608.

Matheson, J. (1968). The value of analysis and computation. *IEEE Transactions on Systems Science, and Cybernetics*, 4:211–219.

McCarthy, J. and Hayes, P. J. (1969). Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4.

Mclaughlin, J. (1987). The utility-directed presentation of graphical simulation. Technical Report TR-87-59, Stanford University.

McNeil, B. J., Pauker, S. G., Sox, H. C., and Tversky, A. (1982). On the elicitation of preferences for alternative therapies. *New England Journal of Medicine*, 306:1259–62.

McNutt, R. and Pauker, S. (1987). Competing rates of risk in a patient with subarachnoid hemorrhage and myocardial infarction: Its now or never. *Medical Decision Making*, 7(4):250–259.

Mesarovic, M., Macko, D., and Takahara, Y. (1970). *Theory of Hierarchical, Multilevel Systems*. Academic Press, New York.

Miller, G. (1956). The magical number seven, plus or minus two. *Psychological Review*, 63:81–97.

Miller, R., McNeil, M., Challinor, S., Masarie, F., and Myers, J. (1986). The INTERNIST-1/Quick Medical Reference project–status report. *Western Journal of Medicine*, 145:816–822.

Miller, R., Pople, E., and Myers, J. (1982). INTERNIST-1: An experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine*, 307:476–486.

Newell, A. and Simon, H. (1963). GPS, a program that simulates human thought. In *Computers and Thought*. McGraw-Hill, New York.

Nicholson, W. (1984). *Microeconomic Theory*. Dryden Press, Chicago.

Nilsson, N. (1986). Probabilistic logic. *Artificial Intelligence*, 28:71–87.

Olmsted, S. (1983). *On Representing and Solving Decision Problems*. PhD thesis, Department of Engineering-Economic Systems, Stanford University.

Owen, D. (1978). The use of influence diagrams in structuring complex decision problems. In Howard, R. and Matheson, J., editors, *Readings on the Principles and Applications of Decision Analysis*, volume II, chapter 38, pages 763–771. Strategic Decisions Group, Menlo Park, Ca.

Papadimitriou, C. and Steiglitz, K. (1982). *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Pauker, S., Gorry, G., Kassirer, J., and Schwartz, W. (1976). Toward the simulation of clinical cognition: Taking a present illness by computer. *American Journal of Medicine*, 60:981–995.

Pearl, J. (1986). Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, 29:241–288.

Pearl, J. (1987). Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32:245–257.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.

Pearl, J. and Verma, T. (1987). The logic of representing dependencies by directed graphs. In *Proceedings AAAI-87 Sixth National Conference on Artificial Intelligence, Seattle, WA*. American Association for Artificial Intelligence, Palo Alto, CA.

Peng, Y. (1986). *A Formalization of Parsimonious Covering and Probabilistic Reasoning in Abductive Diagnostic Inference*. PhD thesis, Department of Computer Science, University of Maryland. TR-1615.

Politser, P. (1984). Explanations of statistical concepts: Can they penetrate the haze of Bayes? *Methods of Information in Medicine*, 23:99–108.

Pople, H. (1982). Heuristic methods for imposing structure on ill-structured problems: The structuring of medical diagnostics. In Szolovits, P., editor, *Artificial Intelligence in Medicine*, pages 119–190. Westview Press, Boulder, CO.

Raiffa, H. (1968). *Decision Analysis: Introductory Lectures on Choice Under Uncertainty*. Addison-Wesley, Reading, Ma.

Reggia, J. and Perricone, B. (1985). Answer justification in medical decision support systems based on Bayesian classification. *Computers in Biology and Medicine*, 15:161–167.

Rivest, R. and Knuth, D. (1973). Bibilography 26: Computing sorting. *Computing Reviews*, 13(6).

Rosenberg, S. (1985). The low-grade non-Hodgkin's lymphomas: Challenges and opportunities. *Journal of Clinical Oncology*, 3:299–310.

Rosenschein, S. and Kaelbling, L. (1986). The synthesis of digital machines with provable epistemic properties. In *Proceedings of the Conference on the Theoretical Aspects of Reasoning About Knowledge*, pages 83–98, Asilomar, CA. AAAI.

Rousseau, W. (1968). A method for computing probabilities in complex situations. Technical Report 6252-2, Center for Systems Research, Stanford University, Stanford, CA.

Russell, S. and Wefald, E. H. (1989). Principles of metareasoning. In Brachman, R. J., Levesque, H. J., and Reiter, R., editors, *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, Toronto. Morgan Kaufman.

Rutledge, G., Thomsen, G., Beinlich, I., Farr, B., Kahn, M., Sheiner, L., and Fagan, L. (1989). Ventplan: An architecture for combining qualitative and quantitative computation. In *Proceedings of the Thirteenth Symposium on Computer Applications in Medical Care, Washington, DC*. IEEE Computer Society Press, Los Angeles, CA.

Samuelson, P. (1973). *Economics*. McGraw-Hill, New York.

Savage, L. (1972). *The Foundations of Statistics*. Dover, New York. 2nd edition, First edition 1954.

Shachter, R. (1986). Evaluating influence diagrams. *Operations Research*, 34:871–882.

Shachter, R. (1988). Probabilistic inference and influence diagrams. *Operations Research*, 36:589–604.

Shachter, R. and Peot, M. (1989). Simulation approaches to general probabilistic inference on belief networks. In *Proceedings of Fifth Workshop on Uncertainty in Artificial Intelligence, Windsor, ON*, pages 311–318. Association for Uncertainty in Artificial Intelligence, Mountain View, CA.

Shafer, G. (1976). *A Mathematical Theory of Evidence*. Princeton University Press, Princeton, NJ.

Sheppard, L. (1980). Computer control of infusion of vasoactive drugs. *Annals of Biomedical Engineering*, 8:431–438.

Sheppard, L. and Sayer, B. (1977). Dynamic analysis of blood-pressure response to hypotensive agents: Studies in post-operative cardiac surgery patients. *Computers and Biomedical Research*, 10:237–245.

Shimony, A. (1955). Coherence and the axioms of confirmation. *Journal of Symbolic Logic*, 20:1–28.

Shortliffe, E. (1982). The computer and medical decision making: Good advice is not enough. *IEEE Engineering in Medicine and Biology Magazine*, 1:16–18.

Shortliffe, E. and Buchanan, B. (1975). A model of inexact reasoning in medicine. *Mathematical Biosciences*, 23:351–379.

Shugan, S. M. (1980). The cost of thinking. *Journal of Consumer Research*, 7:99–111.

Shwe, M., Middleton, B., Heckerman, D., Henrion, M., Horvitz, E., Lehmann, H., and Cooper, G. (1990a). A foundation for normative decision making in internal medicine: A probabilistic reformulation of QMR. Technical Report KSL-90-09, Medical Computer Science Group, Knowledge Systems Laboratory, Stanford University, Stanford, CA.

Shwe, M., Middleton, B., Heckerman, D., Henrion, M., Horvitz, E., Lehmann, H., and Cooper, G. (1990b). A probabilistic reformulation of the Quick Medical Reference System. In *Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care, Washington, DC*. IEEE Computer Society Press, Los Angeles, CA.

Simon, H. (1955). A behavioral model of rational choice. *Quarterly Journal of Economics*, 69:99–118.

Simon, H. (1969). *The Sciences of the Artificial*. M.I.T. Press, Cambridge, Massachusetts.

Simon, H. (1972). The theory of problem solving. *Information Processing*, 71:261–277.

Simon, H. (1973a). The organization of complex systems. In Pattee, H., editor, *Hierarchy Theory*, pages 3–27. G. Braziller.

Simon, H. (1973b). The organization of complex systems. In Pattee, H., editor, *Hierarchy Theory*, pages 3–27. G. Braziller.

Smith, D. (1986). Controlling inference. Technical Report STAN-CS-86-1107, Computer Science Department, Stanford University.

Spiegelhalter, D. and Knill-Jones, R. (1984). Statistical and knowledge-based approaches to clinical decision support systems, with an application in gastroenterology. *Journal of the Royal Statistical Society*, 147:35–77.

Suermondt, H. and Cooper, G. (1988). Updating probabilities in multiply connected belief networks. In *Proceedings of Fourth Workshop on Uncertainty in Artificial Intelligence*, Minneapolis, MN. American Association for Artificial Intelligence.

Suermondt, H. and Cooper, G. (1989). Initialization for the method of conditioning in Bayesian belief networks. Technical Report KSL-89-61, Medical Computer Science Group, Knowledge Systems Laboratory, Stanford University, Stanford, CA.

Szolovits, P. (1982). Artificial intelligence in medicine. In Szolovits, P., editor, *Artificial Intelligence In Medicine*, pages 1–19. Westview Press, Boulder, CO.

Teach, R. and Shortliffe, E. (1981). An analysis of physician attitudes regarding computer-based clinical consultation systems. *Computers and Biomedical Research*, 14:542–558.

Tribus, M. (1969). *Rational Descriptions, Decisions, and Designs*. Pergamon Press, New York.

Tversky, A. and Kahneman, D. (1974). Judgment under uncertainty: Heruistics and biases. *Science*, 185:1124–1131.

Velez-Garcia, E., Durant, J., Gams, R., and Bartolucci, A. (1983). Results of a uniform histopathological review system of lymphoma cases: A ten-year study of the southeastern cancer study group. *Cancer*, 52:675–679.

von Neumann, J. and Morgenstern, O. (1947). *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ.

Warner, H., Toronto, A., Veasy, L., and Stephenson, R. (1961). A mathematical approach to medical diagnosis: Application to congenital heart disease. *Journal of the American Medical Association*, 177:177–183.

Watson, S. and Brown, R. (1978). The valuation of decision analysis. *J.R. Statist. Soc. A.*, 141(1):69–78.

Waugh, N. and Norman, D. (1965). Primary memory. *Psychological Review*, 72:89–104.

Weiss, J., Kulikowski, C., Amarel, S., and Safir, A. (1978). A model-based method for computer-aided medical decision-making. *Artificial Intelligence*, 11:145–172.

Wellman, M. (1988). *Formulation of Tradeoffs in Planning Under Uncertainty*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA.

Zadeh, L. (1983). The role of fuzzy logic in the management of uncertainty in expert systems. *Fuzzy Sets and Systems*, 11:199–227.