N93-32134

Decision-Theoretic Control of EUVE Telescope Scheduling*

Othar Hansson and Andrew Mayer

Heuristicrats Research Inc.
1678 Shattuck Avenue, Suite 310
Berkeley, CA 94709-1631

Computer Science Division
University of California
Berkeley, CA 94720

Abstract

This paper describes DTS, a *decision-theoretic scheduler* designed to employ state-of-the-art probabilistic inference technology to speed the search for efficient solutions to constraint-satisfaction problems. Our approach involves assessing the performance of heuristic control strategies that are normally hard-coded into scheduling systems, and using probabilistic inference to aggregate this information in light of features of a given problem.

BPS, the Bayesian Problem-Solver [8], introduced a similar approach to solving single-agent and adversarial graph search problems, yielding orders-of-magnitude improvement over traditional techniques. Initial efforts suggest that similar improvements will be realizable when applied to typical constraint-satisfaction scheduling problems.

1 DTS Problem Domain and Representation

The Decision-Theoretic Scheduler, DTS, is designed for over-subscribed project scheduling problems. Although our work has not focused on problem representation, the probabilistic techniques used in DTS suggest the possibility of representing stochastic domains, in which, for example, task durations are variable and possibly inter-correlated. Primarily, work on DTS has focussed on search control, particularly through the combination of heuristic evaluation functions. As discussed below, this makes DTS a promising approach for new domains in which sophisticated domain-specific heuristic functions have not been developed. Finally, the utility-theoretic basis of DTS' optimization criteria makes it an attractive approach for problems in which complex tradeoffs must be made among competing tasks and expensive real-world and computational resources.

The current DTS effort is specifically targetted toward experiment scheduling on orbiting telescopes. The initial application domain is the Extreme Ultraviolet Explorer (EUVE) [6], which observes in the wavelength range of 70 to 760 angstroms. The EUVE is operated by the NASA Goddard Space Flight Center and the Center for EUV Astrophysics (CEA) at the University of California, Berkeley. Although the remainder of the paper is concerned with the methodology underlying the system, we describe the problem briefly here.

The tasks in the EUVE scheduling problem are astronomical observations. Although an initial EUVE total sky survey employs fairly short observations, the observations we are concerned with are fairly long. In practice, however, observations will be broken into approximately 30 minute chunks by a variety of unavoidable and largely unpredictable interruptions.

The resources in this scheduling problem are observational instruments. Although the EUVE has several on-board instruments, we concentrate on scheduling guest observations, which are restricted to the EUV spectrometer instrument. Thus, we may consider this a single-resource scheduling problem.

The constraints in the problem are determined by the positions of observational targets, the position of the Observer platform, and the position of obstacles such as planets, the sun and atmospheric anomalies. There are few explicit inter-task constraints, aside from those derived from the time required to retarget the instrument.

1.1 Problem Representation

We phrase these scheduling problems in the language of constraint-satisfaction. Formally, a constraint-satisfaction processing (CSP) problem consists of a set of variables together with a set of constraints on the legal values of those variables. The CSP problem is

*This research was supported by the National Aeronautics and Space Administration under contract NAS2-13340.

C-5

solved when the variables have been instantiated to a set of values that violate none of the constraints. A wide variety of problems can be phrased as CSP problems, including scheduling, graph-coloring, interpretation of visual scenes, etc. (van Hentenryck [23] provides a survey).

A large class of scheduling problems can be represented as constraint-satisfaction problems, by representing attributes of tasks and resources as variables. Task attributes include the scheduled time for the task (start and end time) and its resource requirements. The primary attribute of resources is availability or accessibility. A schedule is constructed by assigning times and resources to tasks, while obeying the constraints of the problem.

Constraints capture logical requirements: a typical resource can be used by only one task at a time. Constraints also express problem requirements: task T_x requires N units of time, must be completed before task T_y, and must be completed before a specified date. Both van Hentenryck [23] and Zweben et al. [26] provide concise illustrative examples of scheduling problems represented as CSP problems. For compatibility and evaluation purposes, our problem representation is based on that of the SPIKE system [15].

1.2 Optimization Criteria

DTS uses multiattribute utility functions to represent user preferences for both solution quality and computational costs. Multiattribute utility theory (MAUT) is a formalized method for quantifying preference relationships among a set of uncertain outcomes. The next section will describe the use of utility functions in search control, but one simple distinction can be made between DTS and traditional scheduling systems at this point. Most scheduling systems have a single vehicle – the heuristic evaluation function – for representing both search control knowledge and user preferences. This overlap of search control and schedule evaluation makes the task of constructing good heuristic functions more difficult than it needs to be.

For example, the ISIS [7] and SPIKE systems [15] employ suitability functions which state the "preferences" of the user as a function of a task and a time assignment. Consider the optimal schedule for a set of tasks. For the SPIKE and ISIS systems, the suitability function which best satisfies the user's preferences is that which "encodes" this optimal schedule by peaked 0/1 values. Such a suitability function essentially encodes search control information. Although this is an extreme case, suitability functions and other heuristic evaluation functions must encode both search control and schedule evaluation information. DTS separates heuristic functions and schedule evaluation, yielding a mathematically principled search algorithm, while at the same time simplifying the knowledge-engineering task for the designer of a new scheduling system.

2 DTS System Overview

There are numerous inadequacies with existing CSP technologies. Existing CSP heuristics, like all heuristic evaluation functions, are imperfect, and exhibit highly domain-specific performance. Although they often provide useful search control advice, they introduce uncertainty into the search algorithms which rely on them. However, CSP problem-solving paradigms do not address this issue because they fail to provide uncertainty models for heuristic information. Consequently, current techniques are forced to pay a large and unnecessary computational price in cases where the heuristic function makes incorrect classifications. Furthermore, the algorithms are doomed to repeat these costly mistakes forever, as there is no learning mechanism designed to improve a CSP heuristic's performance over time.

Existing heuristic functions confuse many different kinds of information. Some heuristic functions estimate the quality of the completion of a partial schedule. Others estimate the difficulty of finding a feasible solution. This confusion results in inadequate guidance for human experts who are charged with developing good heuristic functions. The development of a good heuristic often amounts to little more than parameter-adjustment to improve performance.

The problem of developing heuristics is compounded by the lack of technological developments which allow evidence from multiple heuristic functions to be combined. For this reason, the selection of appropriate heuristics and problem-solving techniques for any given CSP domain remains a craft despite years of comparative study.

DTS, which is derived from previous work on BPS (the Bayesian Problem-Solver) [8], is designed to address these problems. One area of innovation is the *heuristic error model*: a probabilistic semantics for heuristic information, based on the concept of conditional probability in statistical decision-theory [11]. Heuristics are interpreted by correlating their estimates with the actual payoffs of problem-solving instances. When a problem is solved, the heuristic error model is updated, adapting it to the problem's specific characteristics. Multiple heuristics are combined by correlating payoffs with a *set* of heuristic estimates. This provides a sound method for combining multiple heuristics.

This section describes the methodology which forms the core of DTS. In addition to the traditional tools developed for scheduling systems, the DTS approach relies heavily on technologies such as multiattribute utility theory [16; 25], Bayesian probabilistic inference [1; 3; 22], information-value theory [14; 19] and Bayesian learning [4; 17].

2.1 Decisions in Scheduling

DTS employs decision-theoretic techniques to guide the search for feasible and efficient schedules. Decision theory and its central maximum expected utility principle describe methods for making decisions when the outcomes of those decisions are uncertain. A scheduling system is just such a decision-maker. The decisions to be made by a scheduling system include:

1. Which portion of the search tree should be explored next?
2. Should search continue, or should the current best solution be output to the user?
3. If an infeasible schedule must be repaired, which set of repairs is best?

When considered in isolation, these decisions seem very difficult. In fact, they are difficult to formulate and solve in a sound and efficient manner. Existing search algorithms make these decisions in an *ad hoc* manner. Our approach is to apply the standard principles of rational decision-making to these decisions.

The theory of expected utility [24] claims that *rational* decision-makers attach *utilities* to all possible outcomes, and when faced with a decision under uncertainty, select that outcome with maximum expected utility. Utility is the subjective assignment of value to potential outcomes, when the exact outcome is uncertain. An extension of utility theory, which describes the behavior of a decision-maker faced with multiple, and possibly conflicting objectives, is multiattribute utility theory (MAUT).

Under certain natural restrictions on the consistency of a sequence of decisions (i.e., the axioms of decision theory), the fundamental theorem of decision theory states that a consistent decision-maker acts as if he were following the dictates of the theory: i.e., a utility function may be constructed to model his preferences and the MEU principle used to reproduce his decisions. Many artificial intelligence researchers have recently turned to decision-theoretic principles in attempting to engineer sound but resource-conscious systems.

2.2 Heuristic Error Models

The fundamental problem with prior work in scheduling is that the semantics of heuristic functions are defined only in terms of performance: heuristics are "magic" parameters that determine the speed of search. Not surprisingly, an expert's effort in development of scheduling systems is often dominated by time spent handcrafting a high-performance heuristic through parameter adjustment. Because the semantics of heuristics are unclear, even the most sophisticated combination and learning mechanisms are limited in their effectiveness.

DTS takes the approach that there are crucial quantities relating to a state in a search tree, i.e., the attributes of the utility function, including the cost of the search performed, whether a solution was found and the attributes which describe the solution quality. If those attributes were known, decision-making would be trivial. In DTS, heuristic evaluation functions are treated as evidence relating to the value of one or more of the utility attributes. We refer to the set of attributes as the *outcome* of that search tree node.

It is apparent that different heuristics serve to measure different attributes of utility (search cost, solution quality, solution probability). For example, a CSP heuristic such as "Most Constraining Variable" is implicitly encoding information about ease of search: a variable which heavily constrains unassigned variables will produce a smaller search tree. This intuition about the heuristic is borne out empirically.

This association between raw heuristic values and utility attributes is referred to as a *heuristic error model*. Briefly, the heuristic error model provides a simple means of infusing domain specific information into the problem-solving process by associating immediately visible features of a state with a belief about the outcome of that state. "Features" of the state S_i are indicated by a heuristic function, $h(S_i)$, and the association with outcome attributes A_i is provided by the heuristic estimate $\Pr\{h(S_i)|A_i\}$.

Learning Heuristic Error Models Historically, nearly all heuristic search algorithms have used the face-value principle of heuristic interpretation, i.e., behaving as if these estimates were perfect. As a result, most existing heuristic search algorithms violate the basic axioms of consistency and rationality in decision-making.

In contrast, DTS will gather statistics to calibrate the heuristic error model over time, as problems are solved. When introducing the system in a new domain, a *prior* probability distribution will be fine-tuned based on "training exercises" with representative problems. This calibration process will improve DTS performance, tailoring it to the characteristics of real-world problems as they are encountered. When the heuristic function is imperfect, DTS will learn a mapping which "corrects" the heuristic to as great a degree as possible. Finally, the DTS learning capability will reduce the burden on human experts to produce highly complex heuristics. Their experience can be encoded as a default initial belief, or *prior probability*.

Combining Heuristics In our initial experiments in scheduling, a primary advantage of the heuristic error model has been the ability to combine multiple heuristics. Artificial intelligence techniques have never offered powerful methods for combining heuristics. A popular approach is to handcraft a composite heuristic which is a linear combination of individual features.

By combining multiple heuristics, DTS isolates measurements of the difficulty and promise of completing potential assignments. Hence, DTS will make use of heuristics which previously have led to inconsistent performance: if there are any easily characterized contexts (in terms of other features) in which the heuristic performs well, DTS will recognize that fact. This context-dependency of heuristic functions has long been recognized in other search applications such as game-playing.

2.3 Use of Heuristic Error Models

The DTS architecture relies on the Bayesian network data structure [18], the primary artificial intelligence tool for representing and reasoning with probabilistic information. The Bayesian network is used to provide information for decisions such as the most promising region of the search tree to expand next, and the most promising schedule extension or modification to choose next. As described below, Bayesian networks can integrate a variety of information in the service of such decisions, including multiple heuristic evaluation functions and the search tree's topology.

The nodes of a Bayesian network are variables which represent the attributes of the domain. The arcs of the network connect dependent variables, representing relationships among domain attributes. Dependencies can be due to functional, causal or correlative relationships among variables. Dependencies between variables in the network are encoded in a modular fashion by specifying a conditional probability distribution for each network node conditioned on the values of its parents. Although most work has centered on the discrete variable case, Bayesian networks can also incorporate continuous variables.

The variables in the DTS Bayesian network are of two types. One type are variables which represent the multiattribute outcomes (e.g., schedule cost, search cost) of legal partial assignments in the CSP problem's state-space. The other type of variables represent the values of heuristic evaluation functions, the primitive feature recognizers of the domain (e.g., the number of remaining values, the degree of the constraint graph). The structure and semantics of the Bayesian network are described in detail in [12].

The dependency structure and parameters in a Bayesian network enable the efficient computation of the *joint probability* of any instantiation of variables. A fundamental theorem of probability theory indicates that from a joint probability distribution, and thus from a Bayesian network, any well-formed probabilistic question (i.e., conditional probability) can be answered, by the application of Bayes' rule and marginalization. Common queries are of the following form:

- The "next-best-test," or most crucial piece of evidence to gather. In search, this corresponds to the area of the search tree which is most crucial to explore next.
- The conditional probability of a variable instantiation, given the available evidence. In search, such probabilities can be used together with a utility function for maximum-expected-utility decision making, including the choice of task assignments, schedule modifications, etc.
- The most likely instantiation of all variables, given the available evidence. In search, this can be used as a simplified "situation assessment" of the state of the search.

2.4 Utility-Directed Selective Search

DTS employs advanced decision theory techniques to direct its search process. Decision theory, together with the probabilistic inference machinery described above, enables DTS to determine the best portion of the search tree to explore next. In addition to the obvious improvements in search efficiency, this facility improves the flexibility of DTS. By altering the utility function provided as input to the system, DTS may be tailored to trade off increased search time for increases in schedule quality, or to produce schedules with different desirable attributes. For reactive scheduling applications, alterations to the existing schedule can be given negative utility, in which case DTS will avoid them where possible.

The essence of decision-theoretic search control is the realization that there is quantifiable value in the acquisition of information. It should be clear that some pieces of information are more valuable than others. In addition, the acquisition of information has costs – in scheduling search, this cost is increased computation time. If these computations squander time and other resources, the solution may be found too late to be of use. If these computations are neglected, a poor solution may be found. However, if these computations are chosen wisely, the system will provide high quality solutions despite limited computational resources. Decision theory has spawned a subfield known as *information value theory* which deals with the issue of *deciding*, at a metalevel, what information to acquire in order to make better decisions at the base level.

Decision-theoretic search control thus involves the isolation of decisions that are made in the course of search, and applying the techniques of decision theory to make *rational* decisions at these choice points. The decisions made in heuristic search include choices among possible search tree expansions and possible heuristic evaluations. DTS applies information value theory by using the maximum expected utility criterion to control its information-gathering search.

Such decisions form the basis of a selective search algorithm that explores the search tree in a nonuniform manner so as to find a high quality solution in as little time as possible. In simple terms, rather than being a "depth-first" or "breadth-first" search, DTS exhibits a "highest-utility-first" search behavior, gathering information most relevant to the decisions that must be made.

An additional benefit of the decision-theoretic search control is that search control and heuristic information are represented in a declarative manner. As different search spaces (e.g., partial schedules, complete but infeasible schedules, etc.) correspond to different decision problems, DTS can be applied to any search space. In the prototype system, the techniques have been applied to search through the space of valid partial schedules.

2.5 DTS Version 1.0

The DTS prototype employed a simplified decision-theoretic control mechanism which was adapted to a conventional backtracking search algorithm: this allowed for controlled experiments on DTS vs. traditional algorithms.

The only search control decisions made in traditional backtracking systems are the selections of which subtrees of the search graph to explore next. Once a subtree is selected (by selecting the next variable or value), it is explored exhaustively unless a solution is found. Such an ordering problem can be viewed as a decision-tree. Figure 1 depicts the choice of ordering two subtrees A and B. A simple theorem [12] shows that the system's expected utility (search time to first solution) is maximized if variables (or values) are ordered by the quantity $P(v)/C(v)$, where $P(v)$ indicates probability of finding a solution in the subtree, and $C(v)$ indicates the cost of searching the subtree (whether or not a solution is found). $P(v)$ and $C(v)$ are attributes of the payoff mentioned above. The experiments described in the next section confirm that once $P(v)$ and $C(v)$ are learned, this rule outperforms traditional backtracking search algorithms which interpret heuristic estimates at face value. This result indicates that decision-theoretic search-control improves overall system performance. A similar analysis can also be performed for iterative improvement [12].

We note here that while heuristics are usually very good at rank-ordering nodes based on either $P(v)$ or $C(v)$ individually, the rank-ordering for the combination is typically incorrect. DTS' heuristic error model corrects for this.

For clarity, we summarize the prototype implementation here. The prototype performs a backtracking search, using the standard optimizations of forward-checking and dynamic search rearrangement. The search is ordered by the expected utility selection criteria $(P(v)/C(v))$ discussed above. The estimates of $P(v)$ and $C(v)$ are derived from the heuristic error model, using traditional CSP heuristics. The heuristic error model is updated during and between trials

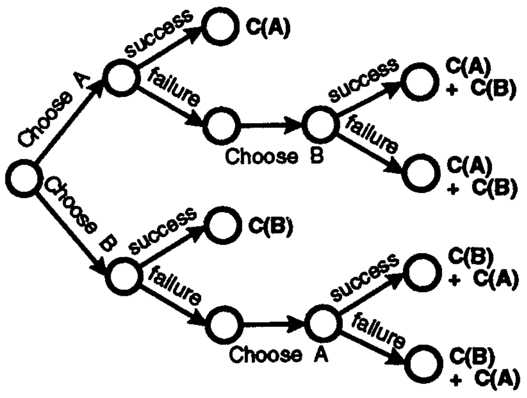

Figure 1: Decision Tree for Value-Ordering Problem (Values A and B)

using a bucketed histogram, and interpreted by Laplacian estimation.

2.6 Performance

Space limits us to a discussion of only three aspects of the DTS prototype's performance characteristics: combination of heuristics, learning heuristic error models, and generalizing learned information.

Combining Heuristics The primary strength of the DTS prototype is the method for combining information from separate heuristic evaluation functions to improve constraint-satisfaction search control. Experiments with the prototype on the Eight Queens and Bridge-Construction Scheduling [23] problems confirm that the combination of heuristic functions provides more information than any of the heuristics taken individually. This translates into significant reductions in overall search time.

Traditionally, CSP algorithms make use of a variable ordering heuristic and a value ordering heuristic. Figure 2 shows the performance of a standard CSP algorithm using all possible pairings (A1, A2, B1, B2) of two well-known variable ordering heuristics (Most Constraining Variable (A), Minimum Domain Variable (B)) and two well-known value ordering heuristics (Least Constraining Value (1), Dechter's Value Heuristic (2)[2]). Also shown is the DTS prototype (DTS-Joint), which dominated the competition by using all four heuristics in combination. The horizontal axis plots the number of problem instances solved and the vertical axis plots the running average of search time over the entire experiment. The plot, but not the average, begins with the tenth problem instance.

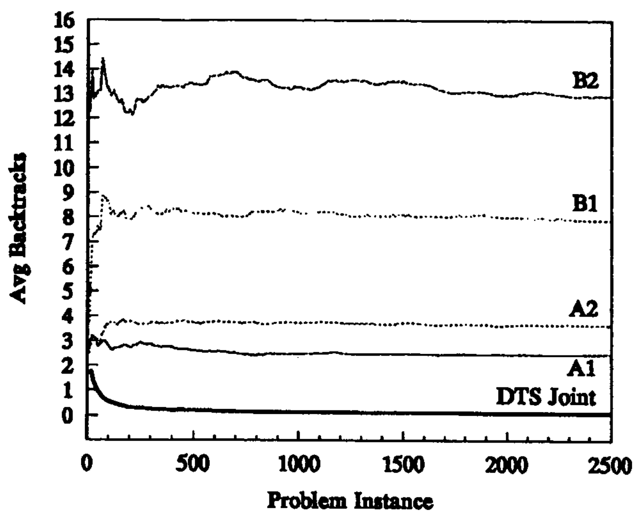

Figure 2: Eight Queens: Combining Heuristics vs. Heuristics in Isolation

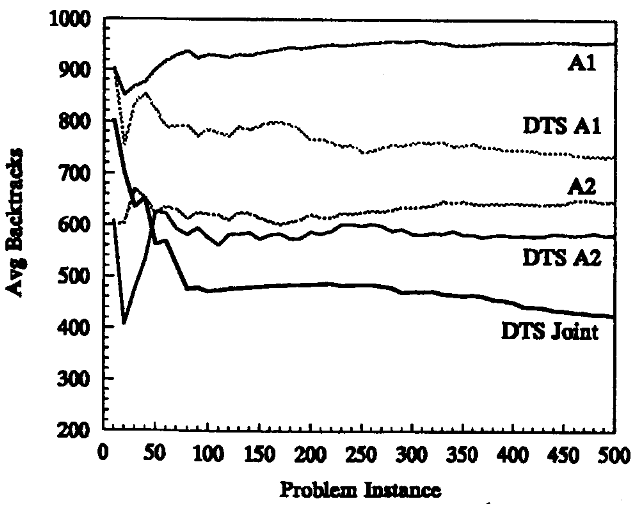

Figure 3: Bridge-Construction Scheduling: Combining Heuristics vs. Heuristics in Isolation

Figure 3 shows a corresponding graph for the Bridge-Construction Scheduling problem. The variable ordering heuristic used was Minimum Domain Variable and the value ordering heuristics were Least Constraining Value (curve A1) and ASAP, "as soon as possible" (curve A2). Also shown are the corresponding individual DTS performance curves (DTS A1, DTS A2) as well as the combined heuristic performance curve (DTS-Joint).

To summarize both graphs, the improvement is seen to be nearly 50% on average for Bridge Construction Scheduling, and over 95% for the Eight-Queens problem. Note that the sharp downward slope of the DTS-Joint *running average* in Figure 3 demonstrates the performance improvement accrued by learning, unattainable using traditional techniques.

Learning Heuristic Error Models Figure 4 displays an example heuristic error model learned over the course of 2500 Eight-Queens problem instances (for the Minimum Domain heuristic). The horizontal axis plots the heuristic function estimate and the vertical axis plots the preference for that estimate. In DTS, preference is based upon the expected utility associated with a heuristic estimate (dashed line). In traditional algorithms, the heuristic is assumed to rank-order alternatives perfectly, and therefore, preference is a monotonic function of the heuristic estimate.

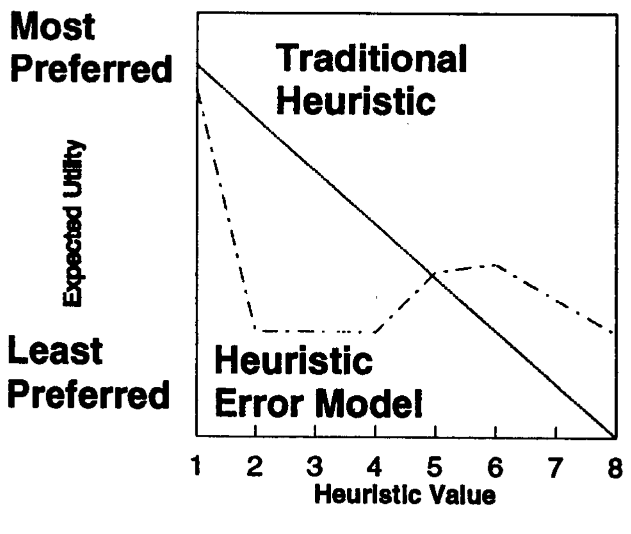

Figure 4: Sample Heuristic Error Model

The discrepancy between the heuristic estimates and the actual utilities explains the poor performance of traditional approaches, which assume perfect heuristic estimates. Further, it explains why DTS outperforms these techniques, as it does not make this assumption, and instead learns to correct for the discrepancy.

Bootstrapping Learned Information An additional benefit of the heuristic error model is the ability to generalize learned data across domains. For example, Figure 5 depicts the performance of DTS on the Thirty-two-Queens problem with 1) no prior heuristic error model, and 2) a heuristic error model generalized (or "bootstrapped") from the 2500 Eight-Queens examples solved in Figure 2. Generalizing data from the simpler domain has reduced search complexity. This is particularly important as the time required to calibrate heuristic error models increases with problem complexity.

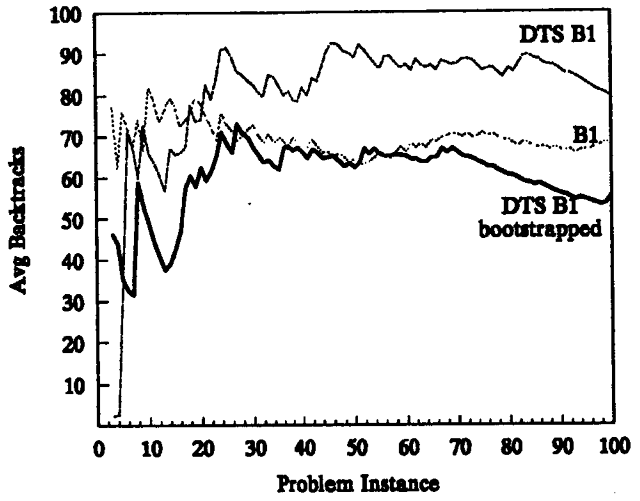

Figure 5: Generalizing Data to Larger Domains

3 Related Work

The DTS system is based on the authors' previous work on the Bayesian Problem-Solver (BPS) system. BPS has been applied to classic AI problem-solving [8], game-playing [10] and planning [9] domains. Similar decision-theoretic approaches are being considered for applications to other complex multiattribute optimization problems.

This work is most closely related, in assumptions and techniques, to the recent work in applying decision theory to problems such as medical diagnosis [13] and image interpretation [5]. Others have applied decision theory to heuristic search applications. These researchers have typically limited themselves to grafting decision-theoretic principles onto existing algorithms[20; 17]. Like the decision-theoretic backtracking discussed above, this makes for an interesting starting point, and there are many more interesting possibilities to explore in the future.

Given its probabilistic basis, this work might be assumed to be related to systems designed for stochastic scheduling problems. Unfortunately, we have not had an opportunity to consider stochastic problems as of yet, although we anticipate that the probabilistic representation and inference mechanisms in DTS will ease the transition to stochastic problems.

4 Conclusions

The use of Bayesian probability theory in DTS underscores that scheduling involves decision-making under uncertainty, and illustrates how imperfect information can be modeled and exploited. The use of multiattribute utility theory in DTS underscores that scheduling involves complex tradeoffs among user preferences. By addressing these issues, DTS has demonstrated promising performance in preliminary empirical testing.

References

[1] R. T. Cox. Probability, Frequency and Reasonable Expectation. *American Journal of Physics*, vol. 14, 1946.

[2] R. Dechter and J. Pearl. Network-Based Heuristics for Constraint-Satisfaction Problems. In *Search in Artificial Intelligence*, L. Kanal and V. Kumar, eds., Springer-Verlag, New York, 1988.

[3] B. de Finetti. *Theory of Probability*. John Wiley, New York, 1974.

[4] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley, New York, 1973.

[5] G. J. Ettinger and T. S. Levitt. An Intelligent Tactical Target Screener. In *Proceedings of the 1989 DARPA Image Understanding Workshop*. Morgan Kaufmann, San Mateo, CA, 1989.

[6] The EUVE Guest Observer Center. *EUVE Guest Observer Program Handbook*. Appendix G of NASA NRA 92-OSSA-5. Center for EUV Astrophysics, Berkeley, January 1992.

[7] M. S. Fox. *Constraint Directed Search: A Case Study in Job-Shop Scheduling*. Pitman, London, 1987.

[8] O. Hansson and A. Mayer. Heuristic Search as Evidential Reasoning. In *Proceedings of the the Fifth Workshop on Uncertainty in Artificial Intelligence*, Windsor, Ontario, August 1989.

[9] O. Hansson, A. Mayer, and S. J. Russell. Decision-Theoretic Planning in BPS. In *Proceedings of the AAAI Spring Symposium on Planning in Uncertain, Unpredictable, or Changing Environments*, Stanford, March 1990.

[10] O. Hansson and A. Mayer. A New and Improved Product Rule. In *Proceedings of the the Eighth International Congress of Cybernetics & Systems*, New York, June 1990.

[11] O. Hansson and A. Mayer. Probabilistic Heuristic Estimates. *Annals of Mathematics and Artificial Intelligence*, 2:209–220, 1990.

[12] O. Hansson and A. Mayer. Decision-Theoretic Control of Artificial Intelligence Scheduling Systems. HRI Technical Report No. 90-1/06.04/5810, September 1991.

[13] D. E. Heckerman. *Probabilistic Similarity Networks.* MIT Press, Cambridge, 1991.

[14] R. A. Howard. Information Value Theory. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SSC-2, 1965.

[15] M. D. Johnston. Knowledge-Based Telescope Scheduling. in *Knowledge-Based Systems in Astronomy*, A. Heck and F. Murtagh, eds., Springer-Verlag, New York, 1989.

[16] R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* John Wiley, New York, 1976.

[17] K.-F. Lee and S. Mahajan. A Pattern Classification Approach to Evaluation Function Learning. *Artificial Intelligence*, vol. 36, 1988.

[18] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, San Mateo, CA, 1988.

[19] H. Raiffa and R. Schlaifer. *Applied Statistical Decision Theory.* Harvard University, 1961.

[20] S. J. Russell and E. Wefald. Principles of Metareasoning. In *Proceedings of the 1st Conference on Principles of Knowledge Representation and Reasoning.* Toronto, 1989.

[21] N. Sadeh. Lookahead Techniques for Activity-Based Job-Shop Scheduling. Technical Report TR CMU-RI-TR-89-2, CMU, the Robotics Institute, 1989.

[22] L. J. Savage. *The Foundations of Statistics.* Dover, New York, 1972.

[23] P. van Hentenryck. *Constraint-Satisfaction in Logic Programming.* MIT Press, Cambridge, MA, 1989.

[24] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior.* Princeton University Press, 1944.

[25] D. von Winterfeldt and W. Edwards. *Decision Analysis and Behavioral Research.* Cambridge University Press, 1986.

[26] M. Zweben, M. Deale and R. Gargan. Anytime Rescheduling. in *Proceedings of the DARPA Planning Workshop*, Morgan Kaufmann, San Mateo, CA, 1990.