19950020975

N95- 27396

S21-63

47271

p-16

Real-time Value-driven Diagnosis

Bruce D'Ambrosio
Department of Computer Science
Oregon State University
dambrosi@research.cs.orst.edu

Abstract

Diagnosis is often thought of as an isolated task in theoretical reasoning (reasoning with the goal of updating our beliefs about the world). We present a decision-theoretic interpretation of diagnosis as a task in practical reasoning (reasoning with the goal of acting in the world), and sketch components of our approach to this task. These components include an abstract problem description, a decision-theoretic model of the basic task, a set of inference methods suitable for evaluating the decision representation in real-time, and a control architecture to provide the needed continuing coordination between the agent and its environment. A principal contribution of this work is the representation and inference methods we have developed, which extend previously available probabilistic inference methods and narrow, somewhat, the gap between probabilistic and logical models of diagnosis.

1 Introduction

In this paper we will present a status report on research into diagnosis as an embedded value-driven activity[1]. Several characteristics become apparent from this perspective that are not normally emphasized in work on diagnosis. First, we view the task as on-going rather than one-shot. Second, we recognize that diagnosis must often be performed under time limitations. Finally, we consider a decision-theoretic model of diagnostic activity in which diagnostic reasoning is not a theoretical activity but a practical one. That is, its goal is not to increase our knowledge, but rather to choose an action which maximizes expected utility.

Four key elements characterize our current approach to these problems. First, we have developed a simple abstract domain which captures these essential elements, which we title the On-Line Maintenance Agent (OLMA) task domain. Second, we have developed a decision-theoretic model of the basic decision task facing an agent in this domain. Third, we have developed probabilistic inference methods suited for solving the resulting decision problem. Fourth, we have developed a problem-solving architecture intended to provide the reactivity needed in embedded, ongoing tasks. We view the third contribution, the development of suitable probabilistic inference mechanisms, as our most significant contribution to date. These methods have been described elsewhere. The goal of this paper is to place them in the context of the larger task of real-time value-driven diagnosis. We begin with a review of the basic task domain, then discuss each of the four elements of our approach. We close with a discussion of related work, limitations, and future directions.

[1]Acknowledgment: This work supported by NSF 91-00530 and the Oregon Advanced Computing Institute.

2 On-Line Equipment Maintenance

Early writings by at least some in AI (e.g., [26]) stress the necessity of recognizing both the limitations and embedded nature of realizable agents. Despite that, much work in AI has proceeded as if the demands placed by the environment are static. In a diagnostic task, for example, the system being diagnosed is often presumed to have broken before diagnosis starts, and to not undergo any further changes (other than those initiated by the diagnostic agent) while diagnosis takes place (an exception is some medical monitoring and diagnosis systems). Further, the environment is presumed to be static enough to allow all solution-quality/time-to-solution tradeoffs to be made at design time (the standard "20 minute solution time" criterion for judging potential expert system applications is a prime example of this). These assumptions, together with assumptions that problem solving proceeds by inference with explicit, categorical representations, characterize much work in AI, whether it is on planning, diagnosis, natural language understanding, or image understanding (although less so in these latter areas).

Diagnosis, in accord with the above, is often formulated as a static, detached process, the goal of which is the assessment of the exact (or most probable) state of some external system. In contrast, we view diagnosis as a dynamic, practical activity by an agent engaged with a changing and uncertain world. We have focused our initial investigations of diagnosis on the task of diagnosing a simple digital system in situ. Our formulation of embedded diagnosis has the following characteristics:

- The equipment [2] under diagnosis continues to operate while being diagnosed.
- Multiple faults can occur (and can continue to occur after an initial fault is detected).
- Faults can be intermittent.
- There is a known fixed cost per unit time the system is malfunctioning.
- The agent senses equipment operation through a set of fixed sensors and one or more movable probes.
- Action alternatives include probing test points and replacing individual components. Each action has a corresponding cost.
- The agent can only perform one action at a time.
- The overall task is to minimize total cost over some extended time period during which several failures can be expected to occur.

We term this task the *On-Line Maintenance* task, and an agent intended for performing such a task an *On-Line Maintenance Agent.* An interesting aspect of this reformulation of the problem is that diagnosis is not a direct goal. A precise diagnosis is neither always obtainable nor necessary. Indeed, it is not even obvious a priori what elements of a diagnosis are even relevant to the decision at hand.

[2] We will use "system" to refer to our diagnostic system, and "equipment" to refer to the target physical system.

3 A decision-theoretic formulation of an OLMA

Our first commitment is that the task is essentially a decision-theoretic one. That is, the essential task of the agent is to *act* in the face of limited information. In order to formulate this problem decision-theoretically, the agent must have knowledge of several parameters of the situation: It must know the cost of each type of replacement or probe act, the cost of system outage, and expected probabilities of component failures over the next decision cycle[3]. A naive attempt to formulate this task decision-theoretically, however, encounters three problems. First, a proper decision-theoretic consideration of this task would require looking ahead over all decisions over the entire operational life of the equipment in order to optimize the first decision. This is clearly computationally intractable. Second, even if the first problem can be solved, time is passing while the agent is computing the first action, and it is not clear how the agent should trade quality of a decision for timeliness of the solution in choosing actions. Finally, the agent must act repeatedly, yet each action is in a new context: not only must a new set of input data be considered, but also a new set of beliefs about system state, based on prior information and computation.

The infinite lookahead, or "small worlds" [23] problem has two subproblems, one for replacement actions and another for probe actions. We circumvent the first subproblem, that of infinite lookahead for replacement actions, as follows. For replacement actions it is possible, under assumptions about stability of the fault transition probabilities and fault costs, to derive off-line an optimal replacement policy conditioned on current beliefs about component states (this would take the form: "replace component x whenever its posterior probability of being faulted is greater than p_x," where p_x depends on parameters of the target system). We use an alternate, equivalent form of this result that requires less a-priori analysis, an assumption of policy stability. This assumption is roughly as follows: If I choose not to replace a component now, then, all other things being equal (ie, no new unexpected sense data), I will make the same choice next time. Under this assumption, the temporal consequences of a decision extend, not for a single sense/act cycle, but several decision cycles into the future. We model the temporal extent of the outcome state for a decision as fixed time-period chosen to satisfy the following constraints:

$$t >> r/f$$
$$t << r/pf$$

where:

t is the outcome state duration (effectively, the multiplier for failure costs),

r is the cost of component replacement,

f is the cost of component failure for a unit clock time, and

p is the probability of component failure during a unit time interval.

The first constraint arises from the observation that, if the cost of replacement is greater than the cost of failure over the outcome period, then our limited look-ahead agent will never

[3]These latter two costs will vary with agent processing capacity, since a slower agent will take longer to make a decision. This will increase the chance of a component failing during a single decision cycle, and increase the cost of a system outage over a decision cycle.

replace a component. The second constraint arises from a more subtle problem: even if we are certain the equipment is functioning correctly now, there is a non-zero probability it will be in failure mode in the outcome state. If the expected cost of that failure ($p * f * t$) is greater than component replacement cost (r), then the agent will always choose to replace, even though it believes the equipment is functioning perfectly. Note that the requirement that both constraints be satisfied limits our approach to equipment with relatively low failure rates.

We resolve the second subproblem, that of determining the expected value of probe actions, by using the standard decision-theoretic heuristic of one-step look-ahead. The result of these two techniques gives us an abstract decision-basis for the first decision as shown in figure 1, where the link from the state at time zero to the value node reflects the costs of operating in that state for one decision time, and the link from the state at time one to the value node reflect the cost of operating in that state for t time units.

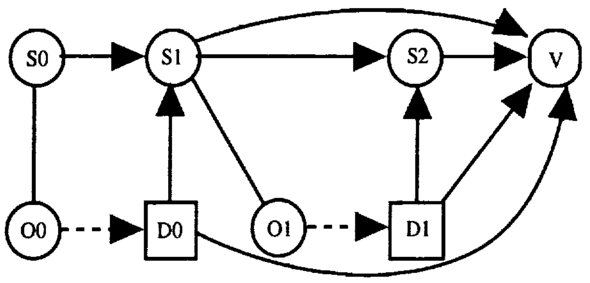


Figure 1: Abstract Decision Basis for first decision

A few notes about this representation are in order. $S0$, $S1$, and $S2$ represent the state of the equipment at times 0, 1, and 2 respectively. The actual state values are unknown to the agent. $O0$ and $O1$ represent the observational data available to the agent at the time decisions $D0$ and $D1$ are to be made. The solid arcs from $S0$ and $S1$ to $O0$ and $O1$ respectively reflect the fact that $O0$ and $O1$ are directly influenced by the (unobservable) system state. The dashed arcs from $O0$ to $D0$ and $O1$ to $D1$ represent the fact that the actual values of these variables will be available at the time the respective decisions will be made. Finally, V represents the value of possible outcomes, and the arcs to it reflect that the value is a function of the actual state of the equipment at times 0 and 1 as well as the direct costs of the actions the agent chooses.

We refer to this as an "abstract" decision basis because we typically will not represent the state of the system or the observational data as single state variables. Rather, we will typically have a structured representation, in the form of a belief net. $S0$, for example, represents the set of unobservable system state parameters and the relationships among them, $O0$ represents the set of observed parameters, and the arc between $S0$ and $O0$ represents the relationship between the unobservable and observable system parameters. This general model can be instantiated with casual or evidential relationships (or both) between unobservable and observable parameters. Consider, for example, a simple sequential two inverter circuit. For that case, we might use causal knowledge to build a specific instantiation of our abstract $S0$ and $O0$ as shown in figure 2. In this figure, "Inv1, P, and Inv2" might all be components of "$S0$", and "I and O" might be directly observable.

Our second problem was that of trading off quality of solution with time to solution. There are two issues here. First, if the equipment is faulted, the longer we delay taking a repair action, the higher the cost incurred. Second, since equipment operation is in parallel with

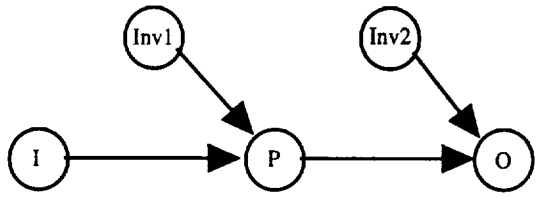


Figure 2: Decision Basis fragment for two inverter system

agent operation, a fault may occur *while* the agent "wastes" time reasoning about a prior, correct set of sense data. We provide no solution to either of these problems here. However, we do present later an incremental (anytime) decision procedure, and sketch experiments we are performing to statistically infer effective reasoning policies.

We resolve our final problem, that of making subsequent decisions, by simply extending the above decision basis forward in time by one decision each cycle. A sample decision basis for the second decision made by the maintenance agent is shown in figure 3. This method would seem to have a problem: one would expect that decision time (and space) would increase at least linearly with time. In fact, both time and space requirements are constant. We defer the discussion of how we accomplish this to the next section, on incremental probabilistic inference.

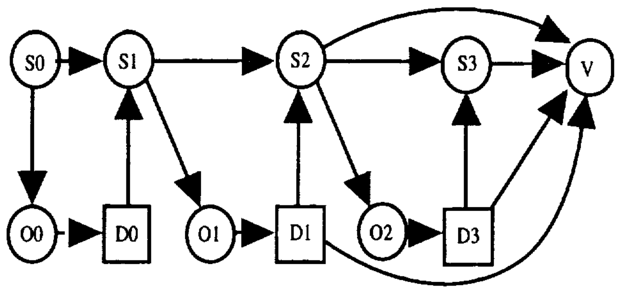


Figure 3: Abstract Decision Basis for second decision

In summary, then, our base-level agent executes the following cycle each time it is called upon to choose an action:

1. Extend the decision basis forward in time by one decision cycle.
2. Acquire current sense data (including probe values).
3. Find the action with minimum expected cost.
4. Post the selected act as evidence in the belief net, prune unneeded old cycles from the net, and return selected action.

4 Incremental Probabilistic Inference

The problem of computing the expected utility for action alternatives can be cast as a belief-net inference problem, as shown by Cooper [3]. However, most current belief net inference facilities provide poor support for the cycle described above. The limitations in existing algorithms are both representational and inferential. The key representational limitation is an inability to represent structure within a single conditional probability distribution. The key inferential problems can be described as lack of *incrementality* with respect to various task requirements.

4.1 Representation

A belief net [21] is a compact, localized representation of a probabilistic model. The key to its locality is that, given a graphical structure representing the dependencies (and, implicitly, conditional independencies) among a set of variables, the joint probability distribution over that set can be completely described by specifying the appropriate set of marginal and conditional distributions over the variables involved. When the graph is sparse, this will involve a much smaller set of numbers than the full joint. Equally important, the graphical structure can be used to guide processing to find efficient ways to evaluate queries against the model. For more details, see [21], [5], [24], [19]. All is not as rosy at it might seem, though. The graphical level is not capable of representing all interesting structural information which might simplify representation or inference. The only mechanism available for describing antecedent interactions in typical general purpose belief net inference algorithms is the full conditional distribution across all antecedents. However, a number of restricted interaction models have been identified which have lower space and time complexity than the full conditional. The noisy-or [21], [22], [13], [14], [1] for example, can be used to model independent causes of an event, and inference complexity is linear in both space and time in the number of antecedents for many inferences. Similarly, various asymmetric [25], [12] and logical relationships are inefficiently represented using a full conditional. Finally, value models used in utility modeling are often factored, for example they may be additive. We have developed an algebraic extension to belief nets which permits conditional distributions to be defined as algebraic compositions of smaller distributions. We have shown that this representation is capable of capturing all known intra-distribution structures, and that simple inference algorithms can make use of this structure to perform inference effectively. Consider, for example, the problem of modelling the conditional distribution of the output of our two inverter circuit given the state of the second inverter and the output of the first inverter. This would normally be modeled as a single monolithic conditional distribution containing (if the inverter has four states: Ok, Stuck-at 0, Stuck-at 1, and Unknown) 16 numbers. The fact that three of the four inverter states are deterministic is lost. However, we model this as follows using our local expression language:

$$\begin{aligned}
exp(O) \quad = \quad & P(O_1|Inv2 = Ok, I1out_0) \\
& + P(O_0|Inv2 = Ok, I1out_1) \\
& + P(O_0|Inv2 = S0) + P(O_1|Inv2 = S1) \\
& + P(O|Inv2 = Uk)
\end{aligned}$$

A total of only six numbers are needed for the above (4 of them are 1.0). But more significantly, we have captured explicitly important structural information such as the fact that the output is independent of the input when the inverter is Stuck-at 1, Stuck-at 0, or Unknown, and that inverter behavior is deterministic in the Ok, Stuck-at 0, and Stuck-at 1 states.

We believe this representation goes a long way towards closing the expressiveness gap between probabilistic and propositional representations for device models.

4.2 Inference

Now that we have an adequate abstract model for the basic on-line maintenance decision and a representation adequate for expressing device and utility models, we need inference methods suited for the task at hand. Despite the fact that current state-of-the-art algorithms exploit the independence information in a belief net to construct efficient computations for probabilistic inference [21], [17], [24], in practice computational cost still grows rapidly [18], limiting application of these techniques to belief nets with a few hundred variables. Also, the services offered by current probabilistic reasoning systems are not well matched to the needs of higher-level problem solvers. As we discussed in [4] and [6], problem solvers typically interleave model construction, revision, and evaluation. As a specific example of this, consider the requirements posed by our formulation of the OLMA task:

- We must extend a model forward in time without discarding all previous inference and starting over. Current methods lack the incrementality with respect to model reformulation operations needed to support this.
- Our decision evaluation methods require that we query arbitrary subsets of the network variables. Current methods lack the incrementality with respect to queries needed to support this.
- The embedded nature of the task requires that we have flexible methods for performing partial inference. Current methods lack the incrementality with respect to completeness needed to support this.

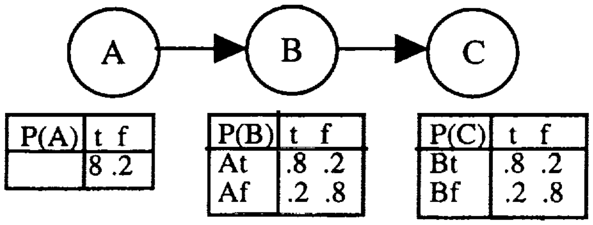
In summary, no existing general-purpose low-level (propositional) probabilistic representation service provides incrementality with respect to model revision and resource usage in a theoretically sound manner. We, of course, have an answer. In this subsection we begin by reviewing the inference problem in a more general setting, and offering a redefinition of the basic inference task. We sketch how an inference engine which performs this task can serve as the core of an incremental probabilistic representation service, and report on the status of our current implementation.

4.2.1 Term Computation

Probabilistic inference in belief nets, as currently defined, is generally taken to be the computation of the prior or posterior marginal, conjunctive, or conditional probability distribution over some subset of variables in a fixed network. This is often an unnecessarily restrictive formulation of the problem. The actual computation of any prior or posterior probability can in general be viewed as a sum over a number of terms (in the extreme case, this occurs as marginalization of the full joint). While the number of terms to be computed is exponential in the size of the network, the time complexity of computation of a single term is linear. Consider the network shown in figure 4.

In this network:

$$\begin{aligned} P(C_t) &= \sum_{A,B} P(C_t|B,A) * P(B) * P(A) \\ &= .8 * .8 * .8 + .8 * .2 * .2 + .2 * .2 * .8 \\ &\quad +.2 * .2 * .2 \\ &= .512 + .032 + .032 + .008 \end{aligned}$$



P(A)	t f
	8 .2

P(B)	t f
At	.8 .2
Af	.2 .8

P(C)	t f
Bt	.8 .2
Bf	.2 .8

Figure 4: Simple Belief Net

We take the computation of a single term as an appropriate primitive task for probabilistic inference, and next show how an inference system with the needed incrementality properties can be built around it.

4.2.2 Desiderata for a TCS

A term computation approach will be interesting only if we can get a significant amount of information through the computation of a small number of terms. While there are many ways this might arise[4], we motivate the approach through the introduction of a critical assumption: we assume that most distributions in a belief net are "asymmetric:"

Definition 1 *A marginal probability distribution is* **Asymmetric** *if one mass element is larger than the remaining element(s). A conditional distribution is Asymmetric if each row satisfies this constraint. In this case it need not be the same element in each row.*

If all the distributions in a belief net are asymmetric, then most of the probability mass for many queries is contained in the first few terms[5]:

Theorem 1 *Given a Belief-net over n two-valued variables such that all distributions are asymmetric with a larger mass at least $(n-1)/n$, then the $n+1$ largest terms in the joint distribution across the variables contain a total mass of at least $2/e$.*

Note that this result is not based on any assumptions about the structure of the network. The degree of asymmetry assumed in the above theorem may seem extreme. However, it is quite natural in many applications, such as failure modeling of complex systems. Thus, our answer to the question of which terms to compute will be to compute the largest terms first.

One could construct a term computation system which merely enumerated elements of the full joint distribution across all variables in a network, as in our example. Indeed, existing proposals for anytime probabilistic inference essentially do this [15], [16]. However, such an approach can be grossly inefficient. There are several sources for this inefficiency: First, there would be a time inefficiency due to unnecessary repetition of sub-computations (eg, the computation of $P(B_t|A_t) * P(A_t)$ in our example). Second, there would be space inefficiency resulting from keeping each term separate. Finally, it is not obvious how such simple methods can be made incremental with respect to newly arriving evidence, queries, or belief net extensions.

Developments in exploiting the probabilistic independence relations expressed in belief nets provide the necessary basis for designing computations which address these problems. In general, the sparser a belief net, the more finely any computation can be partitioned into

[4]For example, through domain dependent knowledge of paradigmatic "cases".
[5]Proof in extended report.

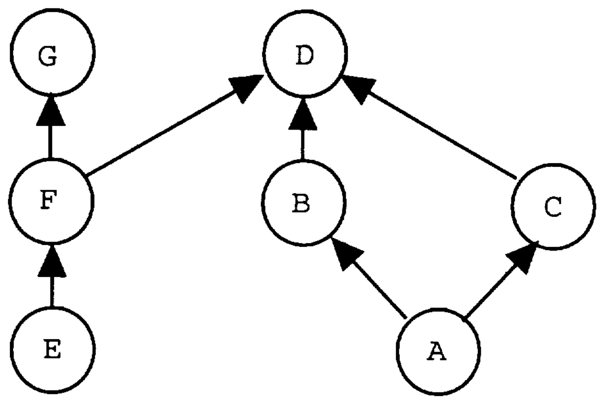


Figure 5: Paradigmatic Belief Net

independent sub-computations which share only a small number of variables. For example, given the net in figure 5, a query for $P(D)$ can be computed by first computing the full joint probability distribution, then marginalizing over all variables except D:

$$\begin{aligned} P(D) &= \sum_{A,B,C,E,F} P(D|B,C,F) * P(F|E) * P(E) \\ &\quad * P(B|A) * P(C|A) * P(A) \end{aligned}$$

However, a much more efficient form of the computation is:

$$\begin{aligned} P(D) &= \sum_{F}(\sum_{E} P(F|E) * P(E)) * \sum_{B,C} P(D|B,C) \\ &\quad * \sum_{A} P(B|A) * P(C|A) * P(A)) \end{aligned}$$

Having done this, we can eliminate redundant computation simply by caching intermediate results. Similarly, we can reduce the space requirement by combining terms when their bindings differ only on variables not needed in the remainder of the computation. In the extreme, each of these can reduce the corresponding complexity (time and space) for computing each term beyond the first from n to $Log(n)$, where n is the number of variables relevant to a query.

In the following section we first develop the basics of term computation (which is inherently incremental with respect to resource consumption) for a static network, set of evidence, and set of queries. We then sketch how the fundamental computation can be made incremental with respect to queries, evidence, and net extension.

4.2.3 Basics of term computation

The elementary primitive out of which we build a term computation system is the construction of a stream of terms for some node in the evaluation poly-tree for a set of queries. This stream will be constructed, recursively, by combining streams of terms from child nodes in the poly-tree. We first describe this evaluation poly-tree and its construction, then explain the term computation process.

Evaluation poly-tree construction Consider the net in figure 5, and assume our queries are for G and D. The expression for $P(D)$ has been given earlier. The expression for $P(G)$ is:

$$P(G) = \sum_F P(G|F) * (\sum_E P(F|E) * P(E))$$

It should be obvious that we can efficiently combine these two expressions into a single evaluation poly-tree:

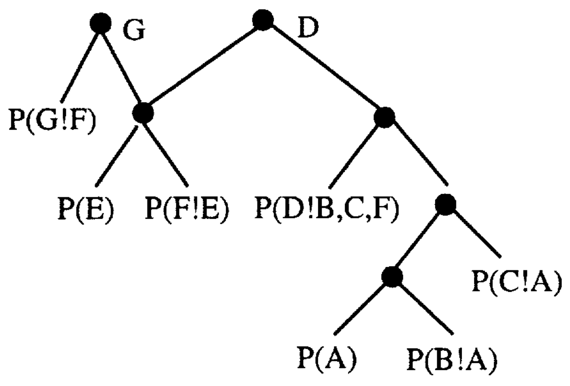


Figure 6: Evaluation Poly-tree for sample query set

Construction of an optimal evaluation poly-tree for an arbitrary query set is a hard problem [19]. However, simple, polynomial-time heuristics perform quite well, and are described in [9], [19]. This previous work was performed in the context of exact query evaluation (that is, computation of all terms), but the theory remains applicable, and so will not be repeated in detail here. The process used in [19] is the basis for our current implementation. That method is essentially a greedy search through the space of partial evaluation trees[6].

Term computation Given an evaluation poly-tree for a query set, the primitive operation at each node in the tree is generation of a term. Term generation is simple: each term is generated by forming the product of a term from the left child and a term from the right child. There are, however, several issues to consider: (1) Control: computations can be performed in either a data-driven or goal-driven fashion; (2) Term selection: the decision of which term to compute next; (3) Term combination: the detection of terms which can be combined. We discuss each of these issues in turn, then demonstrate computation of a term using our example poly-tree.

Control Most probabilistic inference systems are data driven. SPI, the system from which the TCS is derived, however, is query driven. We retain this query driven computational style in our TCS, although it could undoubtably be adapted for data-driven computation. Computation therefore starts at one of the roots of the poly-tree when the problem solver asks for refinement of the distribution for that node, and the root queries its immediate children for terms as needed.

[6]In fact, the situation is slightly more complex when the use of local expressions is considered, since then the set of variables needed below a node in the poly-tree varies depending on the values of the variables instantiated so far. For this reason we actually interleave poly-tree construction and search.

Term selection We earlier stated that we would attempt to minimize the number of terms computed by computing largest terms first. We use AO* to search for the largest term. AO* requires two measures, a measure of "distance so far" and a heuristic estimate of remaining distance. We use the mass computed so far as the inverted "distance traveled so far," and the partial value returned by a partial subterm as our heuristic estimate. This is an admissible heuristic, and so guarantees that the largest term will be in front of the agenda upon termination[7]. Problem solver guidance can be provided in the form of a "scaling function" which has access to term bindings and can scale the probability masses before they are used to order the search agenda.

Term combination (marginalization) The number of terms computed in response to any query is exponential in the number of relevant variables. However, the major advance offered by recent developments in probabilistic inference is reduction of the exponent for computation of complete distributions from number of relevant variables to number of relevant variables in a single factor or cluster, as we discussed earlier. We should not have to pay a higher price simply to achieve incrementality. We can achieve this efficiency by merging completed terms which are distinguished only by bindings on variables not needed at higher levels of the evaluation poly-tree. This creates two problems. First, a term which has already been incorporated into streams at higher levels in the evaluation poly-tree can suddenly have its value change (positively). Simple dependency tracking mechanisms suffice to record the information needed to update these higher terms. Second, exactly what does the AO* guarantee now mean? In poly-tree nodes where marginalization takes place, a partial term can be extended in two ways: by multiplying its value by terms from remaining distributions, or by adding additional ground terms[8]. While we use a heuristic which is admissable in its estimate of the effect of the former, our heuristic is inadmissable with regard to the latter (because it ignores marginalization). This means we can only make a relatively weak statement about terms in streams generated from poly-trees containing marginalization: that the first term returned will be that term whose lower bound is highest after considering all complete ground subterms computed so far. Note that the term need not be "complete" in the sense that further ground terms may be added into it during later computation. It is, however, complete in the sense that it is a sum of a set of complete ground terms.

Complexity The key assumptions we make are that: (1) the probability distributions are sufficiently asymmetric and; (2) the graphical structure of the belief net is sufficiently sparse. Under these assumptions, the evaluation poly-tree will be such that the total number of terms computed in all streams, in the course of computing the first n term requests for each query in the query set, will be n times the number of nodes in the poly-tree. Since the poly-tree is a binary tree, this in turn is $2n$ in the number of variables relevant to the query set. All the operations we have described are either constant time, linear, or at worst $nlog(n)$ (reordering the agendas) in the number of terms in an agenda. Therefore, the total complexity, in the admittedly most optimistic case, is $2n^2log(n)$ where n is the number of variables relevant to a query set and the number of terms requested. Our experience in actually applying this procedure to three tasks, computation of marginal probabilities, most likely composite hypotheses,

[7]This selection criterion is similar to the techniques used by deKleer [11] and Henrion [15]. Both use search on restricted classes of networks for the diagnostic task of finding most likely composite hypotheses, with good results. One contribution of our work is to show how this technique can be used in a more general setting.

[8]A "ground" term is one with a unique binding for each variable in the subtree rooted by the poly-tree node under consideration.

and complete decision analysis, confirms that this estimate is in fact realistic for a typical class of belief nets describing decision models for diagnosis and control of simple digital circuits.

4.2.4 Making Term Computation Incremental

The basic process sketched above is incremental with respect to computation of additional terms for a static query set. We have made this process incremental with respect to queries, evidence, and limited model reformulations (namely the extension and pruning operations needed to support the OLMA task). We omit discussion of these issues here due to lack of space. For a more complete discussion see[7].

4.2.5 Discussion

We have sketched a process which is basically little more than heuristic search for the set of bindings across a set of variables that maximizes the posterior probability across those variables. In another context, deKleer has referred to this as the "Most Likely Composite Hypothesis" problem [10], Henrion has described an algorithm for diagnosis in very large knowledge bases [15], and Pearl has discussed the problem of "Distributed Revision of Composite Beliefs" [20]. From another perspective, Horvitz *et al* have been developing bounded conditioning as an approach to anytime probabilistic inference [16], and Boddy has proposed an anytime approach to dynamic programming [2]. We believe the contributions of our work are several: (1) We have shown how, with caching and marginalization, an incremental probabilistic inference system based on computation of individual terms can be made as efficient at computing all terms (within a factor of $log n$) as the best algorithms for exact inference; (2) We have demonstrated that this process can be made incremental with respect to queries, evidence, and model revisions; (3) We have argued that such a system can serve as the basis for a tractable general-purpose low-level representation service. In particular, this general formulation can be used for evaluation of decision bases, as will be discussed in the next section.

4.3 Value-Driven Diagnosis

There is still a missing link in our base-level OLMA story. What does term computation have to do with evaluating a decision basis, and what does that have to do with diagnosis? Our hypothesis is that the theorem we presented for general *asymmetric* belief nets holds for typical decision bases. Further, we make a stronger hypothesis: that the computation of the few largest terms in the expected utility will be sufficient to distinguish the highest expected utility action. An expected utility is the sum of a number of terms, each of which is a probability of the system resting in a certain outcome state (given the local expression language, perhaps a partial state) multiplied by the (partial)[9] utility of that (partial) state. Our hypothesis will be valid, then, when most terms will be for either low probability outcomes or low utility aspects of an outcome. In our system, then, diagnostic reasoning is driven by the search for high payoff (either positive or negative) utility terms, which drives a search for high probability outcomes, which in turn drives a search for high probability current state information (note that, due to the factoring which takes place in the evaluation poly-tree construction and the structure made explicit in the local expression language, it is not obvious to state a priori what current state information will be queried during expected utility term computation.).

[9] As a result of the additive value model.

Does all this work? We have only executed the system on small digital circuits to date, due to high overhead in our prototype implementation. However, we have observed linear term computation times over a range of 1 to 16 gates. By contrast, exact evaluation of the decision basis exhausts available space (90mb swap on a Sparcstation 1+) at 8 gates. FIgure 7 is a typical performance graph. The task is monitoring a simple 4 gate "half-adder." The vertical axis is cost-per-failure (total scenario cost divided by number of failures), and the horizontal axis is the number of cpu seconds corresponding to one "tick" of the simulation clock (thus, higher numbers simulate a faster cpu for the agent). The solid dots record the cost of running an exhaustive decision algorithm, the crosses record the cost using the TCS. These preliminary results indicate the TCS approach is considerably more robust with respect to task timeliness requirements than is exact computation.

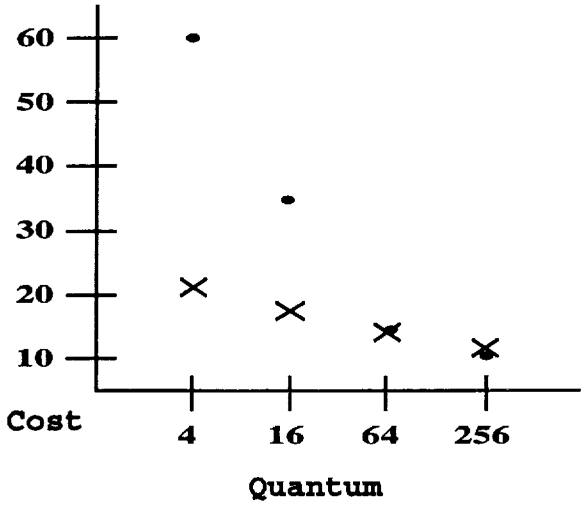


Figure 7: Performance of TCS vs Complete eval

5 Reaction and deliberation in real-time diagnosis

Each TCS point in the above graph is obtained by first performing a series of runs to determine the optimal number of terms to compute for a given cpu speed, then measuring the cost using that number of terms. Thus, the number of terms computed is higher for quantum 256 (1024 terms) than for quantum 1 (1 term). Table 8 shows the optimal number of terms for each quantum.

This presumes that the number of terms computed on each decision cycle is constant and independent of both internal and external state. It further presumes the agent will ignore any sense data that arrives while a decision computation is in progress. This is a very naive approach to real-time problem solving. More generally, we can imagine a meta-decision process which is invoked whenever either: (1) new sense data arrives; or (2) the state of decision evaluation changes (eg, completion of the computation for an additional term). Whenever a state change occurs in either, this meta-decision process has the following options: take an action directly, initiate, terminate, or abort a decision evaluation, or push a new decision evaluation on the stack of current evaluations. Information available for this decision includes the current sense data and the current state of the computation (number of terms computed

Quantum	Optimal Number of Terms	Avg Cost/Failure
1	1	32.5
2	32	28.0
4	32	23.0
8	32	21.5
16	64	20.0
32	256	19.5
64	256	19.3
128	512	16.0
256	1024	14.5

Figure 8: Performance of term computation as Quantum varies

so far, most likely composite hypothesis, etc). For further details of this architecture, see[8]. We are currently applying statistical techniques identify optimal single-level control policies for the OLMA domain.

6 Discussion

It is interesting that the search technique we present should work at all in a decision-making context. There are good theoretical reasons for expecting that search should work well for diagnosis, that is, for identifying the most likely composite hypothesis across component states. The technique proposed here essentially first explores single fault hypotheses, then double fault, and so on. The extension to decision models might seem much less clear. However, recent theoretical results by Druzdel [?] show that almost all Bayesian networks, even those not possessing the local skewness properties described earlier, will contain a few dominant terms. Our experimental results seem to confirm this: The decision networks we use contain many variables whose distributions do not meet the requirements of skewness. Especially noteworthy are the decision and value nodes. Decision nodes, for example, contain no distribution at all, and so provide no search guidance. Nonetheless, our methods provide robust decision-making across a range of quality/time tradeoffs. It may be that this is in part due to the forgiving nature of the problem: there are no catestrophically bad decisions, the worst the agent can do is to merely delay in making the correct decision, and accrue a small failure penalty. It is important to note, however, that even in the worst case (only one term computed) the agent repaired every fault within a few cycles.

We are aware of several limitations in our current approach. Our model of time is quite naive and limited. Our current inability to handle continuous variables is a serious restriction. But perhaps the most severe limitation is the assumption of a complete base level decision model applicable to all situations. We are currently beginning work on both of these latter problems. Specifically, we are exploring methods by which situated decision models can be dynamically constructed from background domain theories, using a meta-level version of the same base level architecture sketched here, and using both bottom up and top-down construction methods.

7 Summary

Diagnosis is often thought of as an isolated task in theoretical reasoning. We have presented a decision-theoretic interpretation of diagnosis as a task in practical reasoning, and sketched components of our approach to this task. The approach makes significant strides to integrating commonly held logical and probabilistic models of diagnosis, as well as incorporating a real-time element. Preliminary results indicate that the approach is significantly more robust in the face of cpu speed variations than traditional approaches.

8 Bibliography

References

[1] M. Agosta. Conditional inter-causally interdependent node distributions ... In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 9–16. Morgan Kaufmann, Publishers, July 1991.

[2] M. Boddy. Anytime problem solving using dynamic programming. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 738–743. AAAI, July 1991.

[3] G. Cooper. A method for using belief networks as influence diagrams. In *Proceedings of the 1988 Workshop on Uncertainty in AI*, pages 55–63. Association for Uncertainty in AI, August 1988.

[4] B. D'Ambrosio. Process, structure, and modularity in reasoning under uncertainty. In *Proceedings of the 1988 Workshop on Uncertainty in AI*, pages 64–72. AAAI, August 1988.

[5] B. D'Ambrosio. Symbolic probabilistic inference. Technical report, CS Dept., Oregon State University, 1989.

[6] B. D'Ambrosio. Incremental evaluation and construction of defeasible probabilistic models. *International Journal of Approximate Reasoning*, July 1990.

[7] B. D'Ambrosio. Term computation systems. Technical report, CS dept., Oregon State University, 1991.

[8] B. D'Ambrosio and T. Fountain. Reaction in real-time decision making. Technical report, CS Dept, Oregon State University, 1991.

[9] B. D'Ambrosio and R. Shachter. Symbolic probabilistic inference. Technical report, Oregon State Univ. CS dept., 1992.

[10] J. de Kleer. Focusing on probable diagnoses. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, pages 842-848. AAAI, July 1991.

[11] J. de Kleer and B. Williams. Diagnosis with behavioral modes. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 1324–1330. IJCAI, August 1984.

[12] D. Geiger and D. Heckerman. Advances in probabilistic reasoning. In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 118–126. Morgan Kaufmann, Publishers, July 1991.

[13] D. Heckerman, J. Breese, and E. Horvitz. The compilation of decision models. In *Proceedings of the Fifth Conference on Uncertainty in AI*, pages 162–173, August 1989.

[14] M. Henrion. Towards efficient probabilistic diagnosis with a very large knowledge-base. In *AAAI Workshop on the Principles of Diagnosis*, 1990.

[15] M. Henrion. Search-based methods to bound diagnostic probabilities in very large belief nets. In *Proceedings of the Seventh Annual Conference on Uncertainty in Artificial Intelligence*, pages 142–150. Morgan Kaufmann, Publishers, July 1991.

[16] E. Horvitz, H. J. Suermondt, and G. Cooper. Bounded conditioning: Flexible inference for decisions under scarce resources. In *Proceedings of the Fifth Conference on Uncertainty in AI*, August 1989.

[17] S. Lauritzen and D. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, B 50, 1988.

[18] Z. Li. Experimental characterization of several algorithms for inference in belief nets. Technical report, Master's thesis, CS Dept., Oregon State University, 1990.

[19] Z. Li and B. D'Ambrosio. An efficient approach to probabilistic inference in belief nets. In *Proceedings of the Annual Canadian Artificial Intelligence Conference*. Canadian Association for Artificial Intelligence, May 1992.

[20] J. Pearl. Distributed revision of composite beliefs. *Artificial Intelligence*, 33(2):173–216, 1987.

[21] J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, Palo Alto, 1988.

[22] Y. Peng and J. Reggia. A probabilistic causal model for diagnostic problem solving - part 1: Integrating symbolic causal inference with numeric probabilistic inference. *IEEE Trans. on Systems, Man, and Cybernetics: special issue on diagnosis*, SMC-17(2):146–162, 1987.

[23] L. Savage. *The Foundations of Statistics.* Dover Publications, 1972.

[24] R. Shachter, B. D'Ambrosio, and B. DelFavero. Symbolic probabilistic inference in belief networks. In *Proceedings Eighth National Conference on AI*, pages 126–131. AAAI, August 1990.

[25] R. Shachter and R. Fung. Contingent influence diagrams. Tech report, Dept. of Engineering Economic Systems, Stanford University, September 1990. In preparation.

[26] H. Simon. *Sciences of the Artificial.* MIT Press, 1974.

a-4.