

# Bayes Network Abstraction for Decision-Theoretic Planning\*

Vu Ha and Peter Haddawy

Decision Systems and Artificial Intelligence Lab

Dept. of EE&CS

University of Wisconsin-Milwaukee

Milwaukee, WI 53201

{vu,haddawy}@cs.uwm.edu

## Abstract

This is a preliminary report on research towards the development of an abstraction-based framework for decision-theoretic planning using Bayesian networks. We discuss two problems: the representation of sets of probability distributions for Bayesian networks, and the issue of representing and abstracting actions using Bayesian networks. For the first problem, we propose the use of *cc-trees* to represent sets of probability distributions and show how propagation in Bayesian networks can be performed without loss of information in this representation. The *cc-tree* representation provides an intuitive and flexible way to make trade-offs between precision and computational cost. For the second problem, we identify a class of planning problems where a simple abstraction technique can be used to abstract a set of actions and to reduce the cost of plan evaluation.

## Introduction

Decision-theoretic approaches to planning, while representationally appealing, tend to be extremely computationally costly. To cope with this complexity, researchers have employed abstraction techniques (Boutilier & Dearden 1994; Doan & Haddawy 1996). Haddawy et al (Haddawy & Suwandi 1994; Doan & Haddawy 1996) identify three types of abstraction that can be used to reduce the complexity of decision-theoretic planning when using a tree-based representation. While the abstraction techniques have been quite successful in reducing the complexity of planning, the tree representation provides little help in structuring domain descriptions. Because of the modularity of representation and the ability to generate explanations afforded by Bayesian networks, we would like to determine whether these abstraction techniques can be mapped onto a Bayes net representation.

---

This work was partially supported by a Fulbright fellowship to Haddawy, by a grant from Rockwell International Corporation, and by NSF grant IRI-9509165.

In this paper, we take a first step in this direction by showing how to abstract sets of actions represented as Bayes net fragments. This work is an extension of our previous work on abstraction (Haddawy & Suwandi 1994; Doan & Haddawy 1996), the theoretical foundations of which can be found in (Ha & Haddawy 1996).

Since our technique for abstracting sets of actions requires the existence of a mechanism for representing and reasoning with sets of probability distributions, we give here a brief review of some existing mechanisms for reasoning with sets of distributions using Bayesian networks. The problem of reasoning with sets of probability distributions has received much attention in the AI research community. Apart from its relevance to abstraction, there are numerous motivations for studying sets of distributions, including attempts to loosen the strictness of the Bayesian paradigm (the *cautious Bayesian*), sensitivity & robustness analysis, and group decision making, to name a few. Previous work on reasoning with sets of distributions in Bayesian networks includes (Fertig & Breese 1989; Tessem 1992; Cano, Delgado, & Moral 1993; Draper 1995; Chrisman 1996). Breese and Fertig (Fertig & Breese 1989), Tessem (Tessem 1992), and Draper (Draper 1995) investigate the problem of reasoning with *interval belief networks (IBN)*, which are generalized Bayesian networks with interval-valued prior/conditional probabilities. It is pointed out by these authors that belief updating in IBNs loses information. More specifically, any direct attempt at propagation (e.g. by extending the computation with interval arithmetic) would almost certainly incur substantial losses of information at each propagation step, eventually leading to vacuous probability bounds. To avoid the information losses in propagating intervals, researchers have introduced alternative representations such as lower probability (Chrisman 1996), or more general representations such as convex polytopes in the probability simplex (Tessem 1992; Cano, Delgado, & Moral 1993). A serious problem with the polytope representation is that the num-

ber of vertices of polytopes can increase exponentially in the propagation process (Tessem 1992), thus rendering the belief updating process intractable in large networks.

In this paper, we propose the use of an alternative representation for sets of distributions: the cc-tree representation<sup>1</sup>. There are several advantages of using the cc-tree representation in the context of abstraction-based planning with Bayes nets. First, this representation is well-suited for decision-theoretic planning purposes: fundamental computations such as plan projection and expected utility computation can be efficiently and elegantly performed using cc-trees. Second, various action abstraction techniques such as *intra-action*, *inter-action*, and *sequential-action* abstraction (Haddawy & Suwandi 1994; Doan & Haddawy 1996) are supported and have intuitive meanings within this representation. These advantages have already been pointed out in (Ha & Haddawy 1996). In this paper, we shall argue that the cc-tree representation *facilitates exact propagation* for sets of probability distributions in interval belief networks, and provides *the flexibility of making tradeoffs, in an anytime manner, between precision and computational cost*.

In the next section, we briefly describe the cc-tree representation and investigate its application to representing sets of marginal/joint probability distributions encoded in interval belief networks. We argue that using cc-trees, belief updating in the networks can be performed without loss of information, and discuss possible schemes for trading off precision for reduction in computational cost. (Due to space limitation, these discussions are very high-level. The rigorous mathematical details are to be reported elsewhere.) We then identify a class of decision-theoretic planning problems where a simple technique can be applied to abstract a set of actions, and use the propagation techniques developed for cc-trees to project abstract actions and plans.

## The CC-Tree Representation

We begin with a description of the terminology. The  $d$ -dimension *Euclidean Space* is the  $d$ -dimension vector space  $R^d$  equipped with the inner product  $\langle \cdot \rangle$ . For a sample space  $\Omega = \{b_1, b_2, \dots, b_d\}$  with cardinality  $d$ , the set of all probability distributions over  $\Omega$  is called

<sup>1</sup>Also called affine-tree in (Ha & Haddawy 1996). We adopt this new name in accordance to the standard terminology in convex geometry theory: cc is the acronym for *convex combination*, or *convex coefficients*. Theoretically, cc-trees encompass polytopes in terms of expressiveness, and can easily be extended to represent non-convex sets of distributions, whenever such need arises (Ha & Haddawy 1997).

the *probability simplex* and denoted by  $\mathcal{S}$ . In the rest of the paper, unless stated otherwise, an interval is implicitly understood as a closed subinterval of  $[0, 1]$ .

**Definition 1 (The CC-Operator)** *The cc-operator  $\otimes$  defined by an interval vector  $\vec{\Lambda} = (\Lambda_1, \dots, \Lambda_n)$  is the function that maps a vector  $\vec{w} = (w_1, \dots, w_n)$  of sets of points in  $R^d$  to the set  $\vec{\Lambda} \otimes \vec{w}$  that consists of all points of the form  $\vec{\lambda} \otimes \vec{x} = \sum_{i=1}^n {}^{cc}\lambda_i x_i$ , where  $\lambda_i \in \Lambda_i, x_i \in w_i, i = 1, \dots, n$ , and  $\sum_i \lambda_i = 1$ .  $\vec{\Lambda} \otimes \vec{w}$  is sometimes written as  $\sum_{i=1}^n {}^{cc}\Lambda_i w_i$ .*

## Illustrations and Examples of the CC-Operator

In the case that all of the intervals  $\Lambda_i$  are the interval  $[0, 1]$ , we denote  $\vec{\Lambda} \otimes \vec{w}$  as  $cc(\vec{w})$ , or  $cc(w_1, \dots, w_n)$ .

- *Polytopes and the cc-operator.* If  $K$  is a polytope with extreme points  $\{x_1, \dots, x_n\}$ , then  $K = \text{conv}(\{x_1, \dots, x_n\}) = cc(\{x_1\}, \dots, \{x_n\})$ .
- *Probability distributions and the cc-operator.* Any distribution  $p \in \mathcal{S}$  can be written as  $p = (p(s_1), \dots, p(s_d)) \otimes (s_1, \dots, s_d)$ , where  $s_i$  represents the  $i$ th vertex of the probability simplex.
- $\otimes$  is closed on  $\mathcal{S}$ . That is, if  $w_i \subseteq \mathcal{S}, i = 1, 2, \dots, n$ , then  $\vec{\Lambda} \otimes \vec{w} \subseteq \mathcal{S}$ .

In the case when each of the sets  $w_i$  is a closed interval of  $R$  (they need not be subintervals of  $[0, 1]$ ), then it can be shown that  $\sum_{i=1}^n {}^{cc}\Lambda_i w_i$  is also a closed interval whose endpoints can be computed using the greedy knapsack algorithm<sup>2</sup>. This special cc-operator sum is denoted by  $\sum_{i=1}^n {}^{icc}\Lambda_i w_i$  (the ‘‘i’’ letter stands for interval). We next briefly describe the algorithm to compute the upper bound of this sum (the lower bound can be computed analogously by symmetry).

Obviously, the upper bound must be achieved by maximizing  $\sum_{i=1}^n {}^{cc}\lambda_i u_i$ , where  $\lambda_i \in \Lambda_i$  and  $u_i$  are upper bounds of  $w_i$ . This is a special case of the continuous knapsack problem: to pack materials from  $n$  different categories into a unit-size knapsack, where material from categories number  $i$  have value  $u_i$  and weight restriction  $\Lambda_i$  (meaning that the weight must be in the interval  $\Lambda_i$ ). The objective is to maximize the total value.

A greedy approach will apparently yield the optimal solution. We first sort the values  $u_i$  in *descending* order, breaking ties arbitrarily. Then proceeding in this order, we try to put into the knapsack as much material of the current category as possible, subjected to two constraints: 1) the weight must be in the constraint interval  $\Lambda_i$  and 2) The sum of the already assigned weights must be less than or equal to the sum

<sup>2</sup>Also called *annihilation/reinforcement algorithm* by Tessem (Tessem 1992).

of the lower bounds of the remaining constraint intervals. The first restriction is explicitly mandatory from the description of the problem, while the second one ensures that the lower bound condition will be met in the future.

The lower bound is computed analogously, using lower bounds of  $w_i$ , and sorting the material in *ascending order* wrt their values.

**Example 1** Let's compute  $\sum_{i=1}^3 {}^{icc}\Lambda_i w_i$ , where  $\Lambda_1 = [.3 .5]$ ,  $\Lambda_2 = [.2 .6]$ ,  $\Lambda_3 = [.1 .7]$   $w_1 = [1 4]$ ,  $w_2 = [0 3]$ ,  $w_3 = [1 2]$ . Thus  $u_1 = 4$ ,  $u_2 = 3$ ,  $u_3 = 2$ , and they are already sorted. First, we assign weight to material number 1, which is computed as  $\min\{.5, 1 - .2 - .1\} = .5$ . The remaining weight is thus  $1 - .5 = .5$ . We then assign weight to material number 2:  $\min\{.6, .5 - .1\} = .4$ . And finally, the weight assigned to the last material is  $1 - .5 - .4 = .1$ , and thus total weight is  $.5 \times 4 + .4 \times 3 + .1 \times 2 = 3.4$ . The lower bound is 0. Thus  $\sum_{i=1}^3 {}^{icc}\Lambda_i w_i = [0 3.4]$ .

**Remarks.** The complexity of the greedy knapsack algorithm is determined mainly by the sorting of the values of the materials.

**CC-Trees** A cc-tree is a rooted, annotated tree where the branches are annotated with intervals and the nodes are annotated with sets of points in  $R^d$ . The only constraint is that for any internal, i.e. non-leaf node  $n$  with children  $\{c_1, \dots, c_k\}$ , the set  $D(n)$  of points associated with  $n$  is the result of applying the cc-operator defined by the intervals  $\Lambda_i$  associated with the branches  $(n, c_i)$  to the sets  $D_i$  of points associated with the children  $c_i$ . Formally, this means that  $D(n) = (\Lambda_1, \dots, \Lambda_k) \otimes (D_1, \dots, D_k)$ . Usually, cc-trees are (consistently and completely) defined by specifying the sets of points associated with their leaves and all the associated intervals. A cc-tree is sometimes interpreted as the set of points associated with its root. One-level cc-trees are also called *cc-stars*.

## Interval Belief Networks

We now illustrate the use of the cc-operator in computing the probability bounds in interval belief networks.

A *Bayesian belief network* is a pair consisting of a directed acyclic graph (DAG) and a collection of prior/conditional probabilities. *Interval belief networks* (Fertig & Breese 1989; Tessem 1992; Draper 1995) are a generalization of Bayesian belief networks where the prior/conditional probabilities are intervals instead of point-values. The usual interpretation of an IBN  $\mathbf{N}$  is that it represents a set of (ordinary) belief networks  $N$ , each of which is obtained from  $\mathbf{N}$  by replacing each interval-valued probability with a consistent point-valued probability, while retaining the

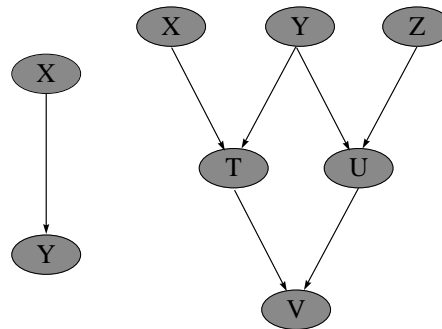


Figure 1: The DAGs of the two examples of interval belief networks

original DAG. The consistency condition requires that the point-valued probabilities must be contained in the corresponding intervals, and must obey the axioms of probability.

**Example.** Consider a DAG with two nodes  $X, Y$  representing two 0/1 binary random variables, and an edge from  $X$  to  $Y$  (Figure 1, left). The link matrices for  $X$  and  $Y$  that specify the prior distribution of  $X$  and the conditional distribution of  $Y$  given  $X$  are given as follows:

$$\mathbf{P}(X) = ([.3, .5], [.5, .7]), \quad \mathbf{P}(Y|X) = \begin{pmatrix} .4 & .6 \\ [.4, .5] & [.5, .6] \end{pmatrix}.$$

A member BN of this IBN is obtained with the following link matrices:

$$P(X) = [.4, .6], \quad P(Y|X) = \begin{pmatrix} .4 & .6 \\ .5 & .5 \end{pmatrix}.$$

The DAG of a Bayesian belief network is in general defined as a *minimal I-map* of the joint distribution it encodes. However, it is obvious that for any IBN  $\mathbf{N}$ , the DAG can only be viewed as an *I-map* for each of the joint distributions that the member belief networks  $N$  represent. The “minimality” property is often violated, because of the extra independency relations that come to surface when a particular combination of point-valued probabilities are used. For example, if we replace the second row of the matrix  $\mathbf{P}(Y|X)$  with  $(.4, .6)$ , then the resulting joint distribution will have an extra independency, namely that  $Y$  is independent of  $X$ . For this distribution, we can drop the edge  $X \rightarrow Y$  in the DAG while still retain its I-mapness. So in a sense, the member “belief networks” are not precisely belief networks, as usually defined and used. Fortunately, this is not a problem in the development of IBNs, as all of the propagation techniques and the relevant theoretical results for Bayesian networks do not

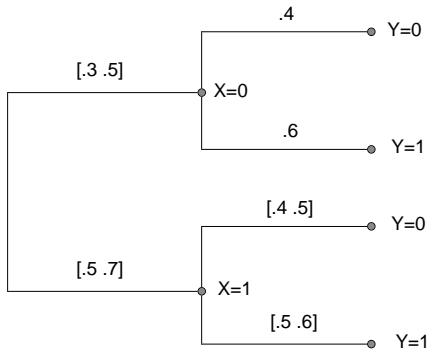


Figure 2: The cc-tree that represents the set of marginal distributions of  $Y$ .

require the minimal I-mapness, but only the I-mapness of the DAGs.

**Computing Probability Bounds in IBNs.** Consider the IBN  $\mathbf{N}$  as described in the previous example. The set of marginal probability distributions of  $Y$  encoded by this IBN can be represented by a 2-level cc-tree (Figure 2). The children of the root are labeled with the values of  $X$ :  $X = 0$  and  $X = 1$ , while the children of  $X = i, i = 0, 1$  are labeled with the values of  $Y$ :  $Y = 0$  and  $Y = 1$ . The branches of the tree are associated with intervals. The branches connecting the root with the nodes  $X = i, i = 0, 1$  are associated with intervals  $\mathbf{P}(X = i)$ , while the branches connecting the nodes  $X = i, i = 0, 1$  and  $Y = j, j = 0, 1$  are associated with intervals  $\mathbf{P}(Y = j|X = i)$ .

Suppose that we are interested in computing the bounds for  $P(Y = 1)$ . This can be done using the greedy knapsack algorithm as follows:

$$\mathbf{P}(Y = 1) = \sum^{cc} \mathbf{P}(X) \mathbf{P}(Y = 1|X) = [.53, .60].$$

It is easy to see that the above method of computing probability bounds can be applied to any causal tree. In a causal tree, any node has at most one parents. So if a node  $X_k$  has ancestors  $X_0, X_1, \dots, X_{k-1}$ , such as  $X_i$  is the parent of  $X_{i+1}, i = 0, \dots, k-1$ , and  $X_0$  is a root node, then the set of marginal probability distributions of  $X_k$  can be represented by a  $(k+1)$ -level cc-tree, and the probability bounds for  $X_k$  can be computed using the recursive greedy algorithm where the depth of the recursion is  $k$ :

$$\begin{aligned} \mathbf{P}(X_k = x_k) &= \sum_{X_0}^{cc} \mathbf{P}(X_0) \sum_{X_1}^{cc} \mathbf{P}(X_1|X_0) \dots \\ &\quad \sum_{X_{k-1}}^{cc} \mathbf{P}(X_k = x_k|X_{k-1}). \end{aligned}$$

In the general case when the DAG of the IBN  $\mathbf{N}$  is a multiply-connected graph, the main idea of computing the probability bounds for any node of the network is to factorize the probability of that node into a series of cc-operator. The factorization can be achieved based on the *join tree propagation algorithm*. We illustrate this idea through an example.

Suppose we have an interval belief network with 6 nodes  $\{X, Y, Z, T, U, V\}$  (Figure 1, right) and suppose we want to compute the probability bounds for  $V$ .

### The Join Tree Propagation Algorithm

• **Input:** A Bayesian network over a set of variables  $X$  and a random variable  $X_i \in X$ .

• **Output:** The marginal distribution  $P(X_i)$  for  $X_i$ .

1. Obtain a family tree for the DAG of the Bayesian network.
2. Assign every node  $X_i$  to one and only one clique containing  $X_i$ . Let  $A_i$  be the set of nodes assigned to clique  $C_i$ .
3. For every clique  $C_i$  define the potential function  $\psi_i(C_i) = \prod_{X_i \in A_i} P(X_i|\Pi_i)$ , where  $\Pi_i$  is the set of parents of  $X_i$ . If  $A_i = \emptyset$  then define  $\psi_i(C_i) = 1$ .
4. Obtain a join tree whose vertices are the cliques  $C_i$ , and mark the clique  $X_i$  is assigned to as the root of the tree (let's denote this clique by  $C_k$ ). The separator set assigned to a branch that connects  $C_i$  and  $C_j$  is  $S_{ij} = C_i \cap C_j$ . The message (to be defined later) that clique  $C_i$  sends to clique  $C_j$  is denoted by  $M_{ij}$  and is a function of the separator set:  $M_{ij}(S_{ij})$ .
5. Compute the joint distribution of the root clique  $C_k$  as:

$$P(C_k) = \psi_k(C_k) \prod_j M_{jk}(S_{jk}),$$

where  $M_{jk}$  is the message that a clique  $C_j$ , which is a child of  $C_k$  in the join tree, sends to  $C_k$ . This message in turn is recursively computed from the messages  $C_j$  receives from its own children as follows. Suppose that  $C_j$  has  $m$  children  $C_{j_1}, \dots, C_{j_m}$ . Then the message  $C_j$  sends to its parent  $C_k$  is computed as

$$\sum_{C_j - S_{jk}} \psi_j(C_j) \prod_{l=1}^m M_{jl}(S_{jl}).$$

6. From the above formula,  $P(X_i)$  can be computed by marginalizing over the random variables in  $C_k - X_i$

Now, we show how this algorithm is applied to compute the marginal probability of  $V$  in the previous example. This yields a decomposition of  $P(V)$  that can be generalized to intervals using the cc-operator.

1. The family tree for the network contains 4 cliques:  $C_1 = XYT$ ,  $C_2 = YTU$ ,  $C_3 = YZU$ ,  $C_4 = TUV$ .
2. The nodes are assigned as follows:  $X, Y, T$  are assigned to  $C_1$ ;  $U, Z$  are assigned to  $C_3$ , and  $V$  is assigned to  $C_4$ . Note that nothing is assigned to  $C_2$ .
3. Compute the potential functions for the cliques.
  - (a)  $\psi_1(XYT) = P(X)P(Y)P(T|XY)$ .
  - (b)  $\psi_2(YTU) = 1$ .
  - (c)  $\psi_3(YZU) = P(Z)P(U|YZ)$ .
  - (d)  $\psi_4(TUV) = P(V|TU)$ .
4. Obtain a join tree for the network. This join tree has 4 nodes  $C_1, C_2, C_3, C_4$  and 3 branches:  $C_1C_2$ ,  $C_2C_3$ ,  $C_2C_4$  with corresponding separators  $S_{12} = YT$ ,  $S_{23} = YU$ , and  $S_{24} = TU$ . The root of this tree is  $C_4$ .
5. The joint distribution for  $C_4 = TUV$  is computed as:

$$\begin{aligned}
P(TUV) &= \psi_4(TUV)M_{24}(TU) \\
&= \psi_4(TUV) \sum_Y \psi_2(YTU)M_{12}(YT)M_{32}(YU) \\
&= P(V|TU) \sum_Y \left( \sum_X P(X)P(Y)P(T|XY) \right) \\
&\quad \left( \sum_Z P(Z)P(U|YZ) \right) \\
&= P(V|TU) \sum_Y P(Y) \left( \sum_X P(X)P(T|XY) \right) \\
&\quad \left( \sum_Z P(Z)P(U|YZ) \right)
\end{aligned}$$

6. Finally, the probability of  $V$  is obtained by marginalizing the above expression over  $T$  and  $U$ :

$$\begin{aligned}
P(v) &= \sum_{T,U} \left( \sum_Y P(Y) \left( \sum_X P(X)P(T|XY) \right) \right. \\
&\quad \left. \left( \sum_Z P(Z)P(U|YZ) \right) \right) \\
&P(v|TU)
\end{aligned}$$

In this example, the probability of node  $V$  is decomposed into a series of 4 cc-operators. The exact probability bounds for  $V$  can then be computed by recursively apply the greedy knapsack algorithm. The depth of this recursion is 4.

It is clear that this method of computing probability bounds can be intractable if the depth of the recursion is large. Techniques such as branch-merging and tree-flattening (Ha & Haddawy 1996) can be used to make tradeoffs between the accuracy of the bounds and reduction of computational cost.

## Abstraction Of Bayesian Networks

In this section, we address the problem of using interval belief networks to abstract two or more Bayesian networks. Let us consider two Bayesian networks  $N_1$  and  $N_2$  over a same set of random variables, with the corresponding DAGs  $G_1$  and  $G_2$ . We are interested in generating an interval belief network  $\mathbf{N}$  that is an abstraction of  $N_1$  and  $N_2$ . The sets of distributions represented by  $\mathbf{N}$  should contain the distributions represented by  $N_1$  and  $N_2$ .

In the case when the two Bayesian networks have the same structure, i.e. when  $G_1 \cong G_2$ , there is a straightforward way to obtain  $\mathbf{N}$ . The DAG of  $\mathbf{N}$  is set to be the same as  $G_1$  and  $G_2$ , while each prior/conditional probability in  $\mathbf{N}$  is set to the smallest interval that contains the corresponding point-valued probabilities in  $N_1$  and  $N_2$ . It follows directly from the definition of IBNs that the resulting IBN will always represent a set of distributions that contains the distributions represented by the original BNs.

In the case when the DAGs  $G_1$  and  $G_2$  are not isomorphic, but share a common topology order<sup>3</sup>, then we can add extra edges until the two DAGs become isomorphic, and the merging technique described above can be applied to these new networks. The adding of new edges proceeds as follows. For each node  $X$ , we take the union of the set of parents of  $X$  in  $G_1$  ( $\Pi_1(X)$ ) and the set of parents of  $X$  in  $G_2$  ( $\Pi_2(X)$ ) to be the new set of parents of  $X$  in both  $G_1$  and  $G_2$ . Note that adding new edges this way does not introduce cycles, and thus the I-mapness of the DAGs is preserved. We also have to update the link matrices for each node in each BN. For each node  $X$ , the associated link matrix  $P_i(X|\Pi_1(X) \cup \Pi_2(X))$  in the network  $N_i$  ( $i = 1, 2$ ) is set to  $P_i(X|\Pi_i(X))$ , using the property that any node in a belief network is independent of its non-descendants, given its parents.

There seems to be little that can be said in the case when there does not exist any topology order common to the two belief networks. Matzkevich and Abramson

<sup>3</sup>Also called *ancestral order*.

(Matzkevich & Abramson 1992) discuss the problem of combining, or fusing belief networks in the context of forcing prior consensus among multiple authors. They give an algorithm for fusing multiple belief networks into a single one (with point probabilities, using some weighting method). They also point out the complexity of this process, due to the significant cost of reversing edges and reorganizing the conditional probability tables. This complexity would almost certainly defeat our initial goal, which is to use abstraction to reduce the cost of plan evaluation. Fortunately, the assumption of having the same topology order often seems reasonable in the context of abstracting actions and plans. This is the issue of the next section

## Action and Plan Abstraction

In this section, we show an application of using interval belief networks to the problem of abstracting temporal probabilistic actions. We will focus on the problem of *inter-action abstraction*, i.e., the problem of grouping together two or more actions into a single abstract action. For planning purposes we require that action abstraction be *sound*, which means that the things we infer from an abstract action must be consistent with the things we infer from its instantiations. So, if we model probabilistic actions by using belief networks, it is natural to model abstract actions by using interval belief networks.

Temporal, probabilistic actions can be modeled using Bayes net fragments (Ngo, Haddawy, & Nguyen 1997). The world is modeled by a temporal belief network having several *slices*, where a slice is a Bayes net fragment that represents the uncertainty about the world at a particular time point. An action is represented by a collection of edges from the nodes in some time slice  $t$  to the nodes in some later time slice  $t'$  ( $t < t'$ ), together with the set of link matrices associated with these edges. Some edges may represent the persistent or intrinsic relationships of the random variables, while some edges specify the effects of the actions.

**Example** Suppose that the world we model consists of four random variables (attributes) called *Rhythm*, *CBF* (cerebral blood flow), *POA* (period of anoxia) and *CD* (cerebral damage) that describe the parameters of a patient. Assume that at any time point, the cerebral blood flow is influenced by the rhythm, and the cerebral damage is influenced by the period of anoxia. The world at time  $t$  is thus modeled with a slice sub-network having the above nodes and two edges:  $Rhythm(t) \rightarrow CBF(t)$  and  $POA(t) \rightarrow CD(t)$ . These links demonstrate the intrinsic relationships among the four attributes at any time point.

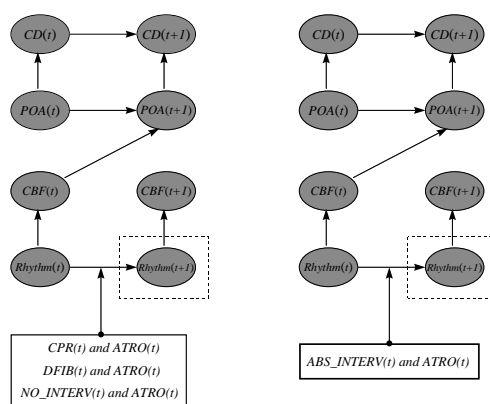


Figure 3: The Bayes net fragments that represent three actions (left) and their inter-action abstraction (right).

Now consider an action that is a combination of intervention/medication treatments on the patient at time  $t$ . The intervention is cardiopulmonary resuscitation (*CPR*), and the medication is the use of atropine (*ATRO*). The effects of this combination are modeled by a 2-slice belief network in figure 3. The link  $CD(t) \rightarrow CD(t + 1)$  specifies the persistence of the attributes *CD*, while the link  $Rhythm(t) \rightarrow Rhythm(t + 1)$  specifies the effects of the treatments.

With the assumption that the intrinsic relationships of the attributes are valid at any time point, we have that any two temporal belief networks that describe two actions have a common topology order: the ordering of the attributes according to any ordering within any time slice, and each attribute at time  $t$  precedes any attribute at any time later than  $t$ . Thus we can model the effects of the abstraction of several actions by an interval belief network.

For example, if we have two other alternative combinations of intervention/medication: *DFIB/ATRO* and *NO\_INTERV/ATRO* (*DFIB*: defibrillation, *NO\_INTERV*: no intervention), each of which only affect the heart rhythm of the patient (but in different ways), then we can group together these three combinations into a single, abstract intervention/medication action, denoted by *ABS\_INTERV/ATRO*. The temporal network for this abstract action will share the same topology with its instantiations, but the link matrix associated with the  $Rhythm(t) \rightarrow Rhythm(t + 1)$  will be an interval matrix resulting from merging the corresponding link matrices of the three concrete intervention/medication actions (figure 3).

## References

- Boutilier, C., and Dearden, R. 1994. Using abstractions for decision-theoretic planning with time constraints. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 1016–1022.
- Cano, J.; Delgado, M.; and Moral, S. 1993. Propagation of convex sets of probabilities in directed acyclic graph. In *Uncertainty in Intelligent Systems*. North Holland. 15–26.
- Chrisman, L. 1996. Propagation of 2-monotone lower probabilities on an undirected graph. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*.
- Doan, A., and Haddawy, P. 1996. Sound abstraction of probabilistic actions in the constraint mass assignment framework. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, 228–235.
- Draper, D. 1995. *Localized Partial Evaluation of Bayesian Belief Networks*. Ph.D. Dissertation, Computer Science Dept., Univ. of Washington, Seattle.
- Fertig, K., and Breese, J. 1989. Interval influence diagrams. In *Proceedings of the Fifth Workshop on Uncertainty in AI*, 102–111.
- Ha, V., and Haddawy, P. 1996. Theoretical foundations for abstraction-based probabilistic planning. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, 291–298.
- Ha, V., and Haddawy, P. 1997. Abstraction-based probabilistic planning: A geometric approach. In preparation.
- Haddawy, P., and Suwandi, M. 1994. Decision-theoretic refinement planning using inheritance abstraction. In *Proceedings, Second International Conference on AI Planning Systems*, 266–271.
- Matzkevich, I., and Abramson, B. 1992. The topological fusion of bayes nets. In *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*, 191–198.
- Ngo, L.; Haddawy, P.; and Nguyen, H. 1997. A modular structured approach to conditional decision-theoretic planning. Under review for uai97.
- Tessem, B. 1992. Interval probability propagation. *International Journal of Approximate Reasoning* 7:95–120.