Efficient Probabilistic Reasoning in Bayes Nets with Mutual Exclusion and Context Specific Independence

Carmel Domshlak DCARMEL@CS.CORNELL.EDU
Computer Science Department
Cornell University
Ithaca, NY 14853, USA

Solomon E. Shimony SHIMONY@CS.BGU.AC.IL
Computer Science Department
Ben-Gurion University
Beer-Sheva 84105, Israel

Abstract

Prior work has shown that context-specific independence (CSI) in Bayes networks can be exploited to speed up belief updating. We examine how networks with variables exhibiting mutual exclusion (e.g. "selector variables"), as well as CSI, can be efficiently updated. In particular, directed-path singly-connected and polytree networks, that have an additional common selector variable, can be updated in linear time (given null and general conjunctive evidence, respectively), where quadratic time would be needed without the mutual exclusion requirement. The above results have direct applications, as such network topologies can be used in predicting the ramifications of user selection in some multimedia data browsing systems.

1. Introduction

Using Bayes networks (BNs) to model uncertain knowledge, and performing inference in this model, are of major interest in both theoretical and applied AI research [20]. As inference over BNs is hard in the general case [5, 6, 24], efficient algorithms for sub-classes of Bayes networks are of extreme importance. Numerous inference algorithms on BNs use a reduction to a known tractable class, in order to perform inference. The reduction is usually exponential in some aspect of the problem instance. Conditioning algorithms [10, 15, 20] use a cutset whose removal reduces inference into a number of (easy) inference problems on polytrees - the number of polytree inference problems is exponential in the cutset size. Similarly, clustering schemes [16, 18] aggregate nodes into macro-nodes organized as a tree, and problem reformulation cost is exponential in the number of nodes in each macro-node.

Relatively recent work has used local independence structure, also known as context-specific independence (CSI) [3] to improve performance of belief updating in Bayes networks [21]. We argue here that in addition to taking advantage of CSI, some application networks exhibit distributions with specific types of mutual exclusion, which can also be used to improve performance. In particular, we examine Bayes networks that are (directed-path) singly connected, except for one additional multiple-valued "selector" variable S (see below for the precise definitions). The problem with having such a "selector" S is that it may have multiple children, creating an arbitrary number of undirected cycles.

Intuitively, a selector variable models user selection from a large set of options. In some user-interface applications, the user can force one of a set of variables into a particular state. System behavior is such that a user action can cause more than one system action. The goal is to predict system behavior, when a distribution over user actions is known. The prediction is necessary in order to achieve better system performance, for example by attempting to optimize actions that are

more likely to be executed in the near future. The unknown user selection is modeled by a selector variable, and the system is modeled by the rest of the network [13].

If the network has n nodes, the selector variable has $O(n)$ possible values. Since the selector may have all network nodes as its children, a natural way to evaluate such a network is by cutset conditioning, with node S being a singleton cutset, resulting in complexity $O(n^2)$ (linear time for each singly connected network problem instance, and $O(n)$ singly connected problem instances - one for each value of S). Other known algorithms can do no better. However, by carefully taking advantage of singly connected network properties, and the CSI and mutual exclusion properties of the selector, probability updating can be done in time linear in the size of the network - the main contribution of this paper.

The rest of the paper is organized as follows. We begin with a formal definition of the problem (network structures, as well as selector variables). The equations and algorithm for computing marginal probabilities for all nodes (also called belief updating) for the null evidence case are then developed. This is followed by extending the results to arbitrary conjunctive evidence, and to cases where the selector variable is not a root node. Finally, an application for the presented results is mentioned, and related work on belief updating is examined, suggesting some future work.

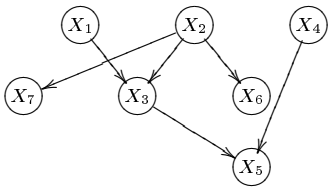
2. Problem Definition

As excellent introductions to Bayes networks abound [4, 19, 20], it suffices to briefly define our notation, as well as to overview the standard inference problems on BNs. A Bayes network $\mathcal{B} = (G, P)$ represents a probability distribution as a directed acyclic graph G (see Figure 1), where its set of nodes V stands for random variables (assumed discrete in this paper), and P, a set of tables of conditional probabilities (CPTs) - one table M_X for each node $X \in V$. For each possible value $x \in D(X)$ (where $D(X)$ denotes the domain of X - the set of possible values for X), the respective table lists the probability of the event $X = x$ given each possible value assignment to (all of) its parents. Thus, the table size is exponential in the in-degree of X. Usually, it is assumed that this in-degree is small - otherwise, representation of the distribution as a Bayes network would not be a good idea in the first place. (We thus assume that the in-degree is bounded by a constant, whenever algorithm runtime results are claimed in this paper.) The joint probability of a complete state (assignment of values to all variables) is given by the product of $|V|$ terms taken from the respective tables [20]. That is, with $Pa(X)$ denoting the parents of X in G, we have:

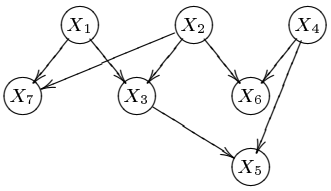
$$P(V) = \prod_{X \in V} P(X|Pa(X)) = \prod_{X \in V} M_X(X|Pa(X))$$

Probabilistic reasoning (inference) is usually in one of two forms: belief updating, and belief revision [20]. In either case, a distinction can be made between a problem with conjunctive *evidence*, which is a partial assignment E to some of the variables (presumably *observed* values for some of the variables), and a reasoning problem with no evidence (or null evidence). The belief updating problem is: compute marginal distributions for all variables given the evidence, i.e. compute $P(X = x|E)$ for all $X \in V$ and for each value $x \in D(X)$. Belief revision, also called most probable explanation (MPE), is finding the assignment A to all the variables that maximizes $P(A|E)$. We discuss only the belief updating problem, although our method and results may also be applicable to belief revision.

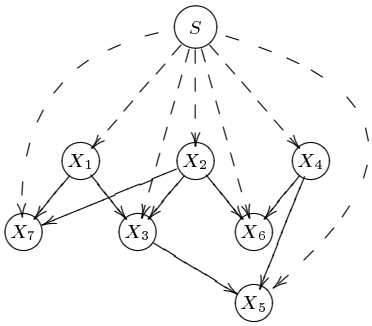
We will be dealing in this paper with specific network topologies and their extensions, defined below. A graph G is directed-path singly connected (DP-singly connected, for short) [23] if for every pair of nodes (s, t) in G, there is at most one *directed* path from s to t (see Figure 1b). The notion DP-singly connected topology is somewhat more general than singly connected (polytree) topology (for which reasoning is known to be easy [17]): The requirement for polytree is that there be at most one path from s to t in the *underlying undirected graph* (see Figure 1a). Clearly, all polytrees are DP-singly connected, but not vice versa. For example, the network in Figure 1b is *not* a polytree, even though it *is* DP-singly connected.



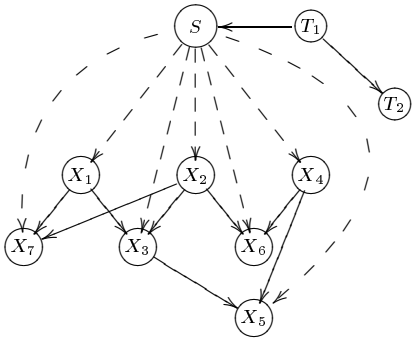
(a) Singly connected (Polytree)



(b) DP-singly connected



(c) DP-singly connected + selector



(d) DP-singly connected + non-root selector

Figure 1: Several topologies of Bayes networks

We consider a class of Bayesian networks arising from applications where the reasoning system needs to address an external event (e.g. a user) making a selection, i.e. forcing *no more than* one of the network variables into a specific state. The a-priori selection distribution, or its dependence on some variables, is known - but for simplicity we initially assume the former. The a-priori distribution is modeled by a selector variable S with no parents. Without loss of generality, let the set of children of S be all the other nodes in the network. The domain of S is the union of the domains of all its children, plus a special value $\perp$. When S has a specific value (some value other than $\perp$), one of its children is forced to the respective value. When S has the value $\perp$, no variable is forced (indicating *no* selection).

Formally, let $\mathcal{B}$ be a Bayes network over the variables $\{X_1, \ldots, X_n, S\}$ with S being a selector variable. We initially assume for simplicity that S is a root node (i.e. has no parents), although this assumption is relaxed later on. We also assume for clarity that the domains of the X_i variables are disjoint, where their values are denoted $x_{i,j}$, with $1 \leq j \leq |D(X_i)|$, respectively. The domain of S is $D(S) = \bigcup_{i=1}^{n} D(X_i) \cup \{\perp\}$, where $S = x_{i,j}$ means that variable X_i is forced to have value $x_{i,j}$, while $S = \perp$ means no variable is forced. The semantics of having the selector as a parent is:

$$P(X_i = x_{i,j} | Pa'(X_i)) = \begin{cases} 1, & S = x_{i,j} \\ 0, & S = x_{i,k},\ k \neq j \\ M_{X_i}(x_{i,j} | Pa(X_i)), & \text{otherwise} \end{cases}$$

where $Pa'(X_i)$ are the parents of X_i in $\mathcal{B}$, and $Pa(X_i) = Pa'(X_i) \setminus \{S\}$. The probabilities $M_{X_i}(x_{i,j}|Pa(X_i))$ are a probability table denoting conditional probabilities of X_i given its parents for the case that S does not select X_i.

Note the mutual exclusion - the selector forces only at most one variable. Also, note the following context-specific independence - if S is known not to select (force) some variable X_i, then X_i becomes

independent of S given the state of the other parents of X_i. Our algorithms takes advantage of both properties. Observe that there is no need to maintain an explicit complete probability table M'_{X_i} for X_i given all its parents $Pa'(X_i)$ - it is sufficient to keep the table for X_i given its parents *excluding* S (for the state $S = \perp$). That is because the explicit array M'_{X_i} would contain numerous identical copies of M_{X_i}.

In this paper we consider belief updating in two classes of BNs with selectors:

- Networks that *excluding* the selectors form *directed-path singly connected* graphs, with *null* evidence.
- Networks that *excluding* the selectors form a *polytree*, with *arbitrary conjunctive* evidence.

For these two problem classes we show how belief updating can be done in linear time, while quadratic time would be needed without the mutual exclusion requirement.

3. Updating for DP-singly-connected+Selector with Null Evidence

Let $\mathcal{B}$ be a Bayes network over the variables $\{X_1, \ldots, X_n, S\}$, with S being a root selector variable as defined above, and such that $\mathcal{B} \setminus \{S\}$ is DP-singly connected (see Figure 1c). Our problem here is to compute the marginal probabilities $P(X_i)$ with no evidence.

Consider any arbitrary node X_i. We distinguish between the three following cases, and analyze them separately:

1. H_i (where S forces (selects) X_i),
2. H_i^+ (where S forces an ancestor of X_i in $\mathcal{B}$),
3. $\overline{H}_i$ (S does not force X_i, nor its ancestors).

Clearly, the hypotheses H_i, H_i^+, and $\overline{H}_i$ constitute a disjoint cover of all possible values of S, and thus:

$$P(X_i = x_{i,j}) = P(X_i = x_{i,j}, H_i) + P(X_i = x_{i,j}|H_i^+)P(H_i^+) + P(X_i = x_{i,j}|\overline{H}_i)P(\overline{H}_i) \quad (1)$$

The case of H_i is straightforward:

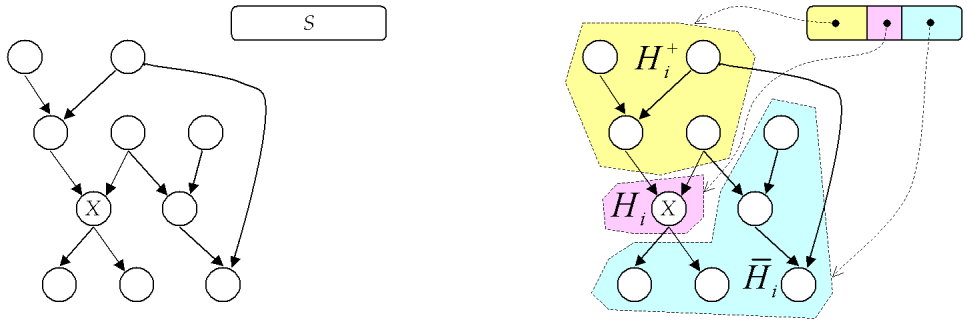
$$P(X_i = x_{i,j}, H_i) = P(S = x_{i,j}, H_i) = P(S = x_{i,j}) \quad (2)$$

where the first equality occurs because once we know H_i, (i.e. that X_i is forced), then $X_i = x_{i,j}$ occurs with probability 1 when $S = x_{i,j}$, and with probability 0 otherwise. The second equality occurs because the event $S = x_{i,j}$ logically entails H_i.

Now consider the case of $\overline{H}_i$. Denote by $D(Pa(X_i))$ the set of all complete assignments (of values to variables) on $Pa(X_i)$, i.e. the cross product of all the domains of the variables $Pa(X_i)$. Observe that (by definition of $\overline{H}_i$):

$$\begin{aligned} P(X_i = x_{i,j}|\overline{H}_i) &= P(X_i = x_{i,j}|S = \perp) \\ &= \sum_{\mathbf{v} \in D(Pa(X_i))} M_{X_i}(x_{i,j}|\mathbf{v}) \prod_{X_m \in Pa(X_i)} P(X_m = \mathbf{v}(X_m)|S = \perp) \end{aligned} \quad (3)$$

where the notation $\mathbf{v}(X_m)$ denotes the element of $\mathbf{v}$ indexed by X_m. Note that the first equality in Equation 3 is due to the fact that, knowing that no ancestors of X_i are forced, the additional



(a) (b)

Figure 2: Selection events on a DP-singly connected Bayes network: (a) A DP-singly connected BN (arcs outgoing from S are omitted); (b) Partition of selection events with respect to the variable X on the BN from (a).

knowledge that *no* node is forced does not further affect X_i. This issue is treated formally later on - see Lemma 1 in the appendix. Evaluating Eq. 3 is equivalent to the standard null-evidence belief propagation for polytree Bayes networks (i.e. passing only π-messages [17]), resulting from conditioning on $S = \perp$.

Finally, consider H_i^+, assuming $P(H_i^+) > 0$. (If $P(H_i^+) = 0$, we do not need to evaluate $P(X_i = x_{i,j}|H_i^+)$ below, as the whole second additive term in Equation 1 becomes 0.) By conditioning on all possible assignments on $Pa(X_i)$ we get:

$$\begin{aligned} P(X_i = x_{i,j}|H_i^+) &= \sum_{\mathbf{v}\in D(Pa(X_i))} P(X_i = x_{i,j}|\mathbf{v}, H_i^+)P(\mathbf{v}|H_i^+) \\ &= \sum_{\mathbf{v}\in D(Pa(X_i))} P(X_i = x_{i,j}|\mathbf{v})P(\mathbf{v}|H_i^+) \\ &= \frac{1}{P(H_i^+)} \sum_{\mathbf{v}\in D(Pa(X_i))} P(X_i = x_{i,j}|\mathbf{v})P(\mathbf{v}, H_i^+) \end{aligned} \quad (4)$$

Observe that the hypothesis H_i^+ can be further decomposed as follows:

$$H_i^+ = \bigcup_{X_m \in Pa(X_i)} (H_m \cup H_m^+)$$

and, since $\mathcal{B} \setminus \{S\}$ is DP-singly connected, the above decomposition is a *disjoint cover*. Hence, for each $\mathbf{v} \in D(Pa(X_i))$, the term $P(\mathbf{v}, H_i^+)$ in Eq. 4 can be decomposed as follows:

$$P(\mathbf{v}, H_i^+) = \sum_{X_m \in Pa(X_i)} P(\mathbf{v}|H_m)P(H_m) \quad + \sum_{X_m \in Pa(X_i)} P(\mathbf{v}|H_m^+)P(H_m^+) \quad (5)$$

The first and the second terms in Eq. 5 distinguish between forcing a parent X_m of X_i, and forcing one of the ancestors of X_m, respectively. Now we rewrite Eq. 4 using the decomposition in Eq. 5, while changing the order of summation, to get:

$$P(X_i = x_{i,j}|H_i^+) = \frac{1}{P(H_i^+)} \sum_{X_m \in Pa(X_i)} \left(\sum_{\mathbf{v} \in D(Pa(X_i))} P(X_i = x_{i,j}|\mathbf{v}) P(\mathbf{v}|H_m) P(H_m) \; + \right. \\ \left. \sum_{\mathbf{v} \in D(Pa(X_i))} P(X_i = x_{i,j}|\mathbf{v}) P(\mathbf{v}|H_m^+) P(H_m^+) \right) \tag{6}$$

The only terms in Eq. 6 that are not trivial are $P(\mathbf{v}|H_m)$ and $P(\mathbf{v}|H_m^+)$. Recall that $\mathcal{B} \setminus \{S\}$ is DP-singly connected, and thus the parents of X_i are independent given H_m, and likewise for H_m^+. Thus we have:

$$P(\mathbf{v}|H_m) \;=\; P(X_m = \mathbf{v}(X_m)|H_m) P(H_m) \prod_{X_k \in Pa(X_i) \setminus \{X_m\}} P(X_k = \mathbf{v}(X_k)|\overline{H}_k)$$

$$P(\mathbf{v}|H_m^+) = P(X_m = \mathbf{v}(X_m)|H_m^+) P(H_m^+) \prod_{X_k \in Pa(X_i) \setminus \{X_m\}} P(X_k = \mathbf{v}(X_k)|\overline{H}_k)$$

All terms can now be computed recursively. To compute marginal probabilities for all variables efficiently, proceed top-down instead, as shown in Figure 3.

DPSC+Selector
Topologically sort the Bayes net $\mathcal{B}$,
  i.e. make $X_m \in Pa(X_i) \Rightarrow i > m$.
for $i = 1$ **to** n **do**
  $P(H_i) = \sum_{x_{i,j} \in D(X_i)} P(S = x_{i,j})$
  $P(H_i^+) = \sum_{X_m \in Pa(X_i)} (P(H_m) + P(H_m^+))$
  $P(\overline{H}_i) = 1 - P(H_i) - P(H_i^+)$
  foreach $x_{i,j} \in D(X_i)$ **do**
    Compute $P(X_i = x_{i,j}|H_i^+)$ using Eq. 6.
    Compute $P(X_i = x_{i,j}|\overline{H}_i)$ using Eq. 3.
    Compute $P(X_i = x_{i,j})$ using Eq. 1.

Figure 3: Algorithm for belief propagation in selector-enhanced DP-singly connected BNs with null evidence.

Theorem 1 *Let $\mathcal{B}$ be a Bayes network over the variables $\{X_1, \ldots, X_n, S\}$ with S a selector variable, such that $\mathcal{B} \setminus \{S\}$ is DP-singly connected with both in-degree and domain for variables $\{X_1, \ldots, X_n, \}$ bounded by a constant. Then algorithm* DPSC+Selector *computes the marginal probabilities $P(X_i)$, for $1 \leq i \leq n$, with no evidence, in time linear in n.*

Proof: Immediate - observe that the algorithm loops n times, and each equation takes constant time to compute, using previously computed terms (assuming in-degree and variable domain size (excluding S) bounded by a constant). Topological sort also takes linear time. □

The complexity of DPSC+Selector is a factor of n better than the optimal respective cut-set conditioning scheme, where the cut-set would be the singleton set $\{S\}$, and where we would need to perform one propagation for each possible state of S. Assuming bounds on the in-degree and domain cardinality allows us to take $O(n)$ as the size of the network. In the standard polytree algorithm,

taking the size of the network to include the size of the CPTs (possibly exponential in the in-degree), allows one to drop these assumptions. Although a similar argument can be made in our case, our algorithm has a further multiplicative factor (the in-degree) in the equations, and we thus cannot quite claim linear time for unbounded in-degree, but can drop the requirement on domain size.

4. Updating in Polytree+Selector with Conjunctive Evidence

Allowing unrestricted conjunctive evidence complicates the computation considerably. For DP-singly connected networks, belief updating with evidence is NP-hard [23], thus we cannot expect to solve the problem in linear time, even *without* the additional selector variable. Observe that evidence at or below "converging" nodes (e.g. X_5, X_7 in Figure 1d), also called *diagnostic* evidence, tends to create further dependencies, thereby also increasing the complexity. Nevertheless, if the network (excluding S) is a *polytree*, efficient belief updating with unrestricted conjunctive evidence is still possible, and can be done in linear time, as shown in this section.

Without loss of generality, we assume that the selector S is a parent of all nodes in $\mathcal{B}$. The belief updating procedure DPSC+Selector for the null evidence case uses a refinement of the Kim and Pearl algorithm [20] that exploits only π-like (i.e. top-down) messages. Here we extend the scheme to work with a refinement of both π and λ messages, as well as additional message types, in order to handle the introduction of evidence.

We begin by introducing some necessary notation on partitions of the set of evidence nodes. For a node X, let U be an arbitrary parent of X, and Y an arbitrary child of X. In order to denote partial evidence, we use notation similar to Pearl's:

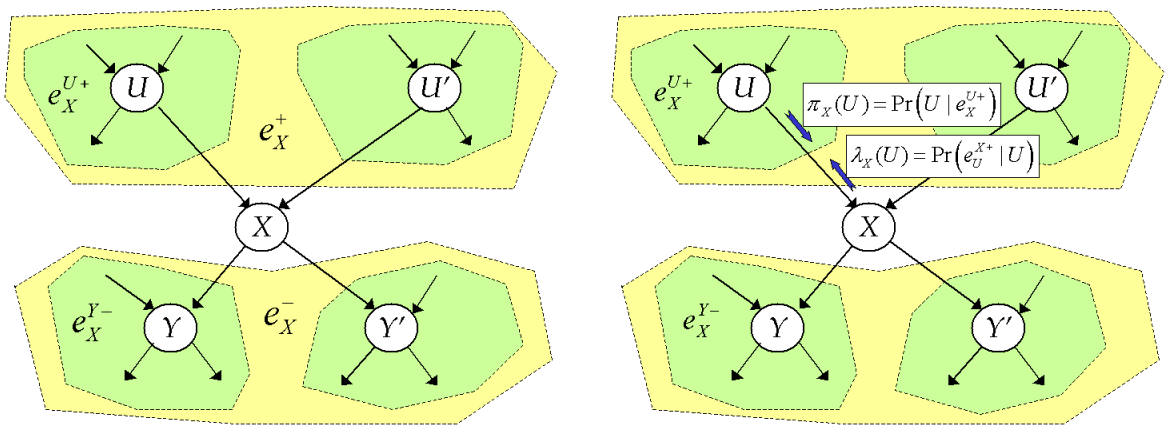
Notation	Meaning
e_X^+	Evidence upstream from X
e_X^-	Evidence downstream from X
e_X^{U+}	Evidence upstream from X via U
e_X^{Y-}	Evidence downstream from X via Y

The above notation for partial evidence is depicted in Figure 4a, while Figure 4b depicts the dynamics of π and λ messages as in [20]. Note that by the term "via U" above, we mean evidence at nodes that are reachable from X in the polytree only through U, which may be either upstream or downstream from U's perspective. Although the terminology we use is standard in [20] and numerous related publications, the term "upstream" (resp. "downstream") does not always conform to one's intuitive understanding of the term. For example, in Figure 4a, evidence at node Y is upstream from node Y', but evidence at node Y' is upstream from Y. That is because the term "upstream" refers only to the direction of the first edge on the path, and in both cases above the first edge is an incoming edge.

In the above specification, we ignore S, which may create some paths in addition to the polytree paths (see the partition in Figure 5b). In order not to complicate the notation even further, following Pearl [20], we assume that all evidence is in specially introduced dummy leaf nodes[1]. These dummy nodes are thus never children of S.

In addition, we need to denote collective selection events. Node S selects some domain value of at most one node in $\mathcal{B}$. Similar to the treatment for evidence (see Figures 5a and 5b):

1. Since each of these dummy nodes have only one parent and no children, and we do not need to compute their marginal distribution, it is sufficient to have each dummy evidence node send a λ message. Specifically, to introduce evidence $X_i = x_{i,j}$, add dummy node X_i' as a child of X_i, sending the message $\lambda_{X_i' \to X_i} = \delta_j$, a vector of dimension $|D(X_i)|$ that is all zeros, except for position j where the value is 1. The only difference on how we treat evidence nodes is in that we have several types of λ messages - but for a dummy node *all* λ messages are equal to δ_j.



(a) (b)

Figure 4: Partial evidence with respect to the variable X (a), and the dynamics of the standard π and λ messages as in [20] (b).

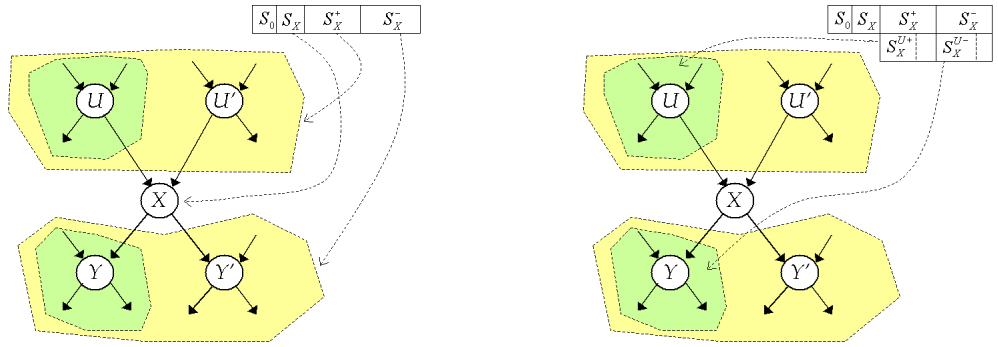
Notation	Denotes the event
S_0	S does not select any variable-value
S_X	S selects (forces) some (undetermined) value of X
$S_{X=x}$	S selects a specific value $x \in D(X)$ for X
S_X^+	Selection of a variable upstream from X
S_X^-	Selection of a variable downstream from X
$S_X^{'-}$	Selection of a variable downstream from X *or* no variable is selected (i.e. $S_X^{'-} = S_0 \bigcup S_X^-$)
S_X^{U+}	Selection of a variable upstream from X reachable in the polytree only through U
S_X^{Y-}	Selection of a variable downstream from X reachable only through Y

Note that since at most one variable is selected and $\mathcal{B} \setminus \{S\}$ forms a polytree, the set $A = \{S_0, S_X, S_X^+, S_X^-\}$ consists of disjoint events. In addition, if the network (without S) is connected, as we assume henceforth, A is a probability space, and thus the sum of probabilities over A is 1. The events denoted with S and its subscript and superscript above are *not* the same as the H notation used in Section 3 for the null evidence case (there, we used H_X^+ to denote selection of a variable that is an *ancestor* of X).

We need to compute $P(X|E)$: for each node X, the probability given the evidence E. Since all evidence is in specially introduced dummy nodes, we can assume without loss of generality that X is not an evidence node. Thus, following [20]:

$$P(X|E) = \frac{P(X, e_X^+, e_X^-)}{P(E)} = \frac{P(e_X^-|X, e_X^+)P(X|e_X^+)P(e_X^+)}{P(E)} = \alpha P(e_X^-|X, e_X^+)P(X|e_X^+) \qquad (7)$$

where $\alpha = \frac{P(e_X^+)}{P(E)}$ is a constant that need not be computed directly - it can be found indirectly by normalizing $P(X|E)$ over all values in the domain of X [20]. Pearl's propagation algorithm now computes the necessary two terms separately, as we do here. We will denote some computed terms



(a) (b)

Figure 5: Illustration of notation for (collective) selection events.

as "messages", in a manner similar to π and λ messages in [20]. However, our notation is extended by requiring additional message types, signifying states of the selector variable. Figure 6 illustrates these message types, formally described below (Figure 4b depicts message types corresponding to π and λ messages as in [20], while Figure 6 illustrates most of our message types). In the table below, we list the messages, their denotation, and their type (scalar, or vector with one element for each value in the domain of the respective variable). The messages which the Kim and Pearl algorithm would use in the polytree resulting from the removal of S, are also pointed out – these have a superscript of 0, denoting that S selects no variable.

Message	Denotes the probability	Type	Equivalent in Kim and Pearl algorithm (if any)
$\pi^0_{U\to X}$	$P(U\|e_X^{U+}, S_0)$	Vector	π message from U to X (their notation: $\pi_X(U)$)
$\pi^+_{U\to X}$	$P(U\|e_X^{U+}, S_X^{U+})$	Vector	
π^0_X	$P(X\|e_X^+, S_0)$	Vector	Cumulative π message at X (their notation: $\pi(X)$)
π^+_X	$P(X\|e_X^+, S_X^+)$	Vector	
π_X	$P(X\|e_X^+)$	Vector	
$\lambda^0_{X\to U}$	$P(e_U^{X-}\|U, S_0)$	Vector	λ message from X to U (their notation: $\lambda_X(U)$)
λ^0_X	$P(e_X^-\|X, S_0)$	Vector	Cumulative λ message at X (their notation: $\lambda(X)$)
$\lambda^-_{X\to U}$	$P(e_U^{X-}\|S_U^{X-}, U)$	Vector	
λ_X	$P(e_X^-\|X, e_X^+)$	Vector	
$\pi^{e0}_{U\to X}$	$P(e_X^{U+}\|S_0)$	Scalar	
$\pi^{e+}_{U\to X}$	$P(e_X^{U+}\|S_X^{U+})$	Scalar	
π^{e+}_X	$P(e_X^+\|S_X^+)$	Scalar	
π^{e0}_X	$P(e_X^+\|S_0)$	Scalar	
σ_X	$P(S_X\|e_X^+)$	Scalar	
σ^+_X	$P(S_X^+\|e_X^+)$	Scalar	
σ^-_X	$P(S_X^-\|e_X^+)$	Scalar	

Finally, we will continue to use the notation M_X to denote the CPT for variable X given no selection, $Pa(X)$ to denote parents of X excluding the selector, and introduce the notation $Ch(X)$ to denote the children (immediate successors) of X.

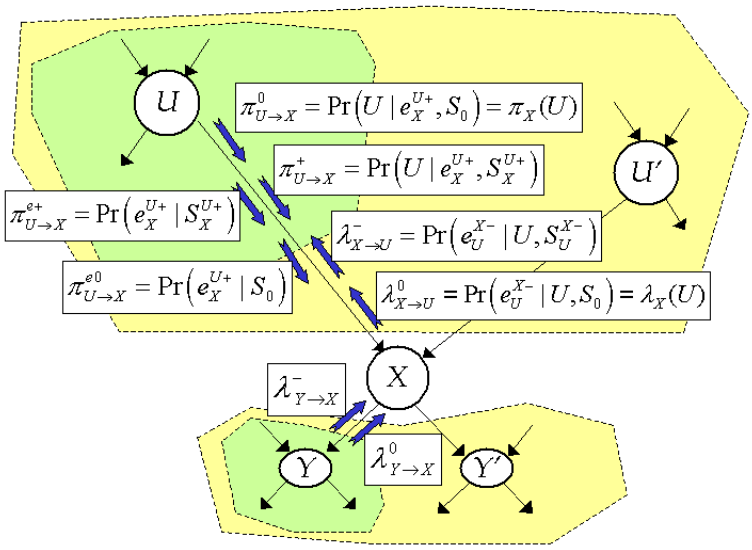


Figure 6: Message types for updating in polytree+selector with conjunctive evidence.

4.1 Computing $\pi_X = P(X|e_X^+)$

The Kim and Pearl algorithm considers all instantiations to parents of X, and then all pieces of upstream evidence are independent due to the polytree property. We cannot do that, due to S. Simply conditioning on S is no good, as we wish to avoid the order n complexity penalty factor (as S has $O(n)$ values). Instead, we condition based on the following partition: $S = S_X \cup S_X^+ \cup S_X'^-$, to get:

$$\begin{aligned} P(X|e_X^+) &= P(X|S_X, e_X^+)P(S_X|e_X^+) + P(X|S_X^+, e_X^+)P(S_X^+|e_X^+) + P(X|S_X'^-, e_X^+)P(S_X'^-|e_X^+) = \\ &\frac{P(S_{X=x})}{P(S_X)}\sigma_X + \pi_X^+\sigma_X^+ + \pi_X^0\sigma_X'^- \end{aligned} \tag{8}$$

where $P(X|S_X'^-, e_X^+) = P(X|e_X^+, S_0) = \pi_X^0$ holds because X is independent of S given e_X^+ when S is known not to select a variable upstream from X (see Appendix for details).

Achieving an effective algorithm requires efficient computation of the above σ messages:

Theorem 2 *The σ messages relation to π messages are as follows:*

$$\sigma_X = P(S_X|e_X^+) = \frac{\pi_X^{e0}P(S_X)}{P(e_X^+)} \tag{9}$$

$$\sigma_X'^- = P(S_X'^-|e_X^+) = \frac{\pi_X^{e0}P(S_X'^-)}{P(e_X^+)} \tag{10}$$

$$\sigma_X^+ = P(S_X^+|e_X^+) = \frac{1}{P(e_X^+)} \sum_{U\in Pa(X)} P(S_X^{U+})\pi^{e+}_{U\to X} \prod_{W\in Pa(X)\setminus\{U\}} \pi^{e0}_{W\to X} \tag{11}$$

where $P(e_X^+)$ is such that $\sigma_X + \sigma_X'^- + \sigma_X^+ = 1$.

For a proof of the above theorem, see Appendix. In addition, we need to evaluate the π messages:

Theorem 3 *The vector π messages can be computed recursively using other π messages and λ messages as follows:*

$$\pi_X^+ = P(X|S_X^+, e_X^+) = \frac{1}{\sigma_X^+ P(e_X^+)} \sum_{U \in Pa(X)} P(S_X^{U+}) \pi_{U \to X}^{e+} \times \left(\sum_{\mathbf{v} \in D(Pa(X))} M_X(X|\mathbf{v}) \pi_{U \to X}^+(\mathbf{v}(U)) \prod_{W \in Pa(X) \setminus \{U\}} \pi_{W \to X}^{e0} \pi_{W \to X}^0(\mathbf{v}(W)) \right) \quad (12)$$

$$\pi_{U \to X}^+ = P(U|S_X^{U+}, e_X^{U+}) = \frac{1}{P(S_X^{U+}) \pi_{U \to X}^{e+}} \prod_{W \in Ch(U) \setminus \{X\}} \lambda_{W \to U}^0 \times \left[\pi_U^{e+} \pi_U^+ P(S_U^+) + \pi_U^{e0} \left(P(S_{U=u}) + \pi_U^0 \sum_{V \in Ch(U) \setminus \{X\}} P(S_U^{V-}) \frac{\lambda_{V \to U}^-}{\lambda_{V \to U}^0} \right) \right] \quad (13)$$

where the latter is based on the partition $S_X^{U+} = S_U \cup S_U^+ \cup (S_U^- \setminus S_U^{X-})$ - see Appendix for details. The messages π_X^0 and $\pi_{U \to X}^0$ were developed in [20] - the equations are repeated below for convenience:

$$\pi_X^0 = \sum_{\mathbf{v} \in D(Pa(X))} M_X(x|\mathbf{v}) \prod_{U \in Pa(X)} \pi_{U \to X}^0 \quad (14)$$

$$\pi_{U \to X}^0 = \beta \pi_U^0 \prod_{W \in Ch(U) \setminus \{X\}} \lambda_{W \to U}^0 \quad (15)$$

with β a normalizing constant, such that the sum of $\pi_{U \to X}^0$ over the domain of U is 1.

Theorem 4 *The scalar π evidence messages can be computed recursively using other π messages and λ messages as follows:*

$$\pi_{U \to X}^{e+} = P(e_X^{U+}|S_X^{U+}) = \frac{1}{P(S_X^{U+})} \sum_{u \in D(U)} \frac{\lambda_X^0(u)}{\lambda_{X \to U}^0(u)} \times \left[\pi_U^{e+} \pi_U^+(u) P(S_U^+) + \pi_U^{e0} \left(P(S_{U=u}) + \pi_U^0(u) \sum_{W \in Ch(U) \setminus \{X\}} P(S_U^{W-}) \frac{\lambda_{W \to U}^-(u)}{\lambda_{W \to U}^0(u)} \right) \right] \quad (16)$$

$$\pi_X^{e+} = P(e_X^+|S_X^+) = \frac{P(S_X^+|e_X^+) P(e_X^+)}{P(S_X^+)} = \frac{1}{P(S_X^+)} \sum_{U \in Pa(X)} P(S_X^{U+}) \pi_{U \to X}^{e+} \prod_{W \in Pa(X) \setminus \{U\}} \pi_{W \to X}^{e0} \quad (17)$$

$$\pi_{U \to X}^{e0} = P(e_X^{U+}|S_0) = \pi_U^{e0} \sum_{u \in D(U)} \pi_U^0(u) \prod_{W \in Ch(U) \setminus \{X\}} \lambda_{W \to U}^0(u) \quad (18)$$

$$\pi_X^{e0} = P(e_X^+|S_0) = \prod_{U \in Pa(X)} P(e_X^{U+}|S_0) = \prod_{U \in Pa(X)} \pi_{U \to X}^{e0} \quad (19)$$

Theorem 5 *The $\lambda^-_{X\to U}$ message, from an arbitrary node X to an arbitrary parent U can be computed as follows:*

$$\lambda^-_{X\to U} = P(e_U^{X-}|S_U^{X-},U) = \frac{\pi^{e0}_X}{P(S_U^{X-})\pi^{e0}_{U\to X}} \sum_{x\in D(X)} \Bigg[P(S_{X=x})\lambda^0_X(x) + \sum_{\mathbf{v}\in D(Pa(X)\setminus\{U\})} M_X(x|\mathbf{v},U) \times \Bigg(P(S_X^-)\lambda^-_X(x) \prod_{Z\in Pa(X)\setminus\{U\}} \pi^0_{Z\to X}(\mathbf{v}(Z)) + \lambda^0_X(x) \sum_{Z\in Pa(X)\setminus\{U\}} \frac{P(S_X^{Z+})\pi^{e+}_{Z\to X}\pi^+_{Z\to X}(\mathbf{v}(Z))}{\pi^{e0}_{Z\to X}} \prod_{W\in Pa(X)\setminus\{U,Z\}} \pi^0_{W\to X}(\mathbf{v}(W)) \Bigg) \Bigg] \quad (20)$$

Theorem 6 *The total message $\lambda^-_X \to U$ can be computed as follows:*

$$\lambda^-_X = P(e^-_X|S^-_X,X) = \frac{1}{P(S^-_X)} \sum_{Y\in Ch(X)} P(S_X^{Y-})\lambda^-_{Y\to X} \prod_{W\in Ch(X)\setminus\{Y\}} \lambda^0_{W\to X} \quad (21)$$

For convenience, the λ^0 messages, derived in [20], appear below:

$$\lambda^0_{X\to U} = P(e_U^{X-}|S_0,U) = \sum_{x\in D(X)} \lambda^0_X(x) \sum_{\mathbf{v}\in D(Pa(X)\setminus\{U\})} M_X(x|\mathbf{v},U) \prod_{Z\in Pa(X)\setminus\{U\}} \pi^{e0}_{Z\to X}\pi^0_{Z\to X}(\mathbf{v}(Z)) \quad (22)$$

$$\lambda^0_X = P(e^-_X|S_0,X) = \prod_{Y\in Ch(X)} \lambda^0_{Y\to X} \quad (23)$$

Finally, the aggregated marginal selector probabilities are computed recursively as follows:

$$P(S_X^{U+}) = P(S_U) + \sum_{W\in Pa(U)} P(S_U^{W+}) + \sum_{V\in Ch(U)\setminus\{X\}} P(S_U^{V-}) \quad (24)$$

$$P(S_U^{X-}) = P(S_X) + \sum_{W\in Pa(X)\setminus\{U\}} P(S_X^{W+}) + \sum_{V\in Ch(X)} P(S_X^{V-}) \quad (25)$$

$$P(S_X^+) = \sum_{U\in Pa(X)} P(S_X^{U+}) \quad (26)$$

$$P(S_X^-) = \sum_{V\in Ch(X)} P(S_X^{V-}) \quad (27)$$

Observe that all equations in this section can be evaluated in constant time, assuming bounded in-degree and bounded domain for all variables (except for S), and that all quantities appearing in the equations can be retrieved in constant time.

4.2 Computing $\lambda_X = P(e^-_X|X,e^+_X)$

In Pearl's algorithm, the term e^+_X can be dropped from the conditioning above, as X d-separates the upstream nodes from the downstream nodes. Unfortunately, we *cannot* do that due to the common ancestor S. Nevertheless, after conditioning on S aggregate states, we *will* be able to drop e^+_X. We partition S into the following two aggregate states: S^-_X, i.e. selection of a variable downstream from

X, and S'^{+}_X, denoting all the rest (i.e. selection at X, upstream of X, or no selection). Thus, by conditioning on the above aggregate states of S, we have:

$$\lambda_X = P(e^-_X|X, e^+_X) = P(e^-_X|S'^{+}_X, X, e^+_X)P(S'^{+}_X|X, e^+_X) + P(e^-_X|S^-_X, X, e^+_X)P(S^-_X|X, e^+_X)$$

However, evidence downstream from X is independent of the evidence upstream from X, given S^-_X and X (due to Lemma 1 from the Appendix). Likewise for the case S'^{+}_X, but additionally, because there is no selection below X in this case, the probability of the downstream evidence is indifferent to the selection, and we might as well have S_0 as the conditioning event. Also, the conditional distribution over the two aggregate states of S must sum to 1, and we get:

$$\begin{aligned} P(e^-_X|X, e^+_X) = P(e^-_X|S_0, X)(1 - P(S^-_X|X, e^+_X)) + P(e^-_X|S^-_X, X)P(S^-_X|X, e^+_X) = \\ \lambda^0_X(1 - P(S^-_X|X, e^+_X)) + \lambda^-_X P(S^-_X|X, e^+_X) = \lambda^0_X + P(S^-_X|X, e^+_X)(\lambda^-_X - \lambda^0_X) \end{aligned} \quad (28)$$

We can break down the term $P(S^-_X|X, e^+_X)$ as follows, using Bayes:

$$P(S^-_X|X, e^+_X) = \frac{P(e^+_X, X|S^-_X)P(S^-_X)}{P(X|e^+_X)P(e^+_X)} = \frac{P(e^+_X, X|S^-_X)P(S^-_X)}{\pi_X P(e^+_X)}$$

where all terms have been derived earlier, except for (using Lemma 1 from the Appendix):

$$P(e^+_X, X|S^-_X) = P(e^+_X, X|S_0) = P(e^+_X|S_0)P(X|e^+_X, S_0) = \pi^{e0}_X \pi^0_X$$

Putting it all together, we get:

$$\lambda_X = P(e^-_X|X, e^+_X) = \lambda^0_X + \frac{\pi^{e0}_X \pi^0_X P(S^-_X)}{\pi_X P(e^+_X)}(\lambda^-_X - \lambda^0_X) \quad (29)$$

Note that the equations hold for strictly positive distributions. For distributions containing zeros, they can still be used, but must be applied carefully. This is a detailed implementation problem, rather than a conceptual problem: observe that if $P(e^+_X) = 0$, then evidence probability is zero and value of posteriors is undefined anyway. Likewise, if we have $\pi_X(x) = 0$ for some x, there is no need to compute the respective $\lambda_X(x)$, as it will be multiplied by zero to get $P(X = x|E) = 0$.

4.3 Polytree+Selector Algorithm

After defining all the message types above, and developing the equations relating them, the algorithm is straightforward. There are at least two ways to present what is essentially the same algorithm: a) "distributed" message passing, as in [20], and b) recursive computation, as in [22], which is the way we elected to present the algorithm. With the latter option, shown in Figure 7, all messages are just notational and algorithmic conveniences - values to be computed and stored. (As opposed to being sent to other nodes, which is what happens in the distributed version).

In the description of the algorithm, by "recursively compute and store" applied to a message m_i (where i denotes that this is a message from variable X_i), we mean the following recursive procedure:

1. Check the slot for the message m_i in the data structure for X_i, and if not empty, return the stored value.

2. Otherwise, use the respective equation for computing the message, by recursively computing and storing any message used in the equation.

3. Store m_i in its respective slot in the data structure for m_i, and return m_i.

Polytree+Selector

for $i = 1$ **to** n **do**
  Initialize a storage data structure for X_i, with a slot for
  all X_i messages, and mark all slots as"empty".
for $i = 1$ **to** n **do**
  Recursively compute and store $\pi_X = P(X_i|e^+_{X_i})$ using equations 8 to 27.
  Recursively compute and store $\lambda_X = P(e^-_{X_i}|X_i, e^+_{X_i})$ using equation 29.
  For all $x_{i,j} \in D(X_i)$ compute $p_j = P(X_i = X_{i,j}|e^+_{X_i})P(e^-_{X_i}|X_i = X_{i,j}, e^+_{X_i})$
    and normalize: $P(X_i = X_{i,j}|E) = \frac{p_j}{\sum_k p_k}$

Figure 7: Algorithm for belief propagation in selector-enhanced polytree BN with conjunctive evidence.

Theorem 7 *Let $\mathcal{B}$ be a Bayes network over variables $\{X_1, \ldots, X_n, S\}$ with S a selector variable, such that $\mathcal{B} \setminus \{S\}$ is a polytree with both in-degree and domain for variables $\{X_1, \ldots, X_n, \}$ bounded by a constant. Let E be arbitrary conjunctive evidence. Then algorithm* Polytree+Selector *computes all marginals $P(X_i|E)$, for $1 \leq i \leq n$, in time linear in n.*

Proof: As argued in the Appendix, the message definitions contain no recursive references, and all base cases are defined. Since the total size of all the messages is linear in n, all messages are computed at most once, and each equation contains a bounded number of terms, the theorem follows. □

Note that in order to avoid cumbersome details, the algorithm as listed handles only strictly positive distributions (other than conditionals given S). In order to apply the algorithm in the presence of zero probabilities, care must be taken when using the equations. In some cases, a zero in a denominator means that the marginals are undefined and an error should be returned to the calling program or user. In other cases, the zero probability in a denominator also entails that a complete expression can be discarded and need not be evaluated at all. Examples of both these types of occurrence can be seen at the end of Section 4.2.

4.4 Extensions

Although the above algorithms are sufficient for one application discussed below, it is certainly of interest whether the scheme is applicable to belief updating in more general topologies, for example, relaxing the assumption that S is a parent of all the other variables, and the assumption that S is a root node. The former is straightforward, and requires only minor adjustments to the equations (namely dropping some terms) - the details are uninteresting. Note, however, that if S is a parent of only $m < n$ nodes, we only gain a factor of m by using our algorithm, rather than a factor of n.

Relaxing the assumption that S is a root node is more interesting, and can be done as long as all the following conditions hold:

1. S together with its non-descendants constitute a polytree T.
2. Every path, from any node in T to any descendent of S, contains S.
3. The subgraph G'' induced by the nodes downstream from S is a polytree, *or* G'' is DP-singly connected and contains no evidence nodes.

In these conditions hold (for example, see Figure 1d assuming the evidence is only at T_1 and/or T_2), the following simple scheme still performs correct belief updating, and takes only linear time:

1. Compute the probability of S given the evidence in T, by (temporarily) removing all descendents of S and performing belief updating on the resulting polytree.

2. Temporarily remove from the original network all nodes except for S and the nodes downstream, and use the algorithm for DP-singly connected+selector if there is no evidence downstream from S, or the algorithm for polytree+selector if downstream evidence exists.

5. Discussion

In this section, we briefly outline some applications for selectors in Bayes networks, related work, and additional possible extensions for the algorithm as issues for future research.

5.1 Applications for our scheme

The issue of selectors in Bayes networks was raised in trying to predict system behavior in sophisticated user-interface type applications, where a user action can cause more than one system action in response. The prediction can help to achieve better system performance by attempting to optimize system actions that are more likely to be executed in the near future. For instance, consider a database of multi-media documents that is accessed by its users *remotely*. Large amounts of information must be delivered from the database to the users quickly, on demand.

To see the issues involved, consider a medical record in which (multi-media) patient information is continuously gathered, and periodically accessed by physicians (e.g., see [14]). Each medical record consists of numerous information components. Additionally, each component may also have several presentation options that are pre-generated and stored in the database. A typical example would be a CT image, which can be presented either plain, or segmented. When tackling the presentation of media-rich information in such systems, one should decide *when and what to present* from the whole content of the document.

In particular, [11, 12] discuss an adaptive system that presents multimedia data items and/or multi-component web pages based on preferential constraints from the author, as well as on user selection. Numerous components (in some cases all components) are visible to the user, either as a placeholder (icon or line of text), or in another form (such as a full picture, a text window, etc.). The user can point and click on any of the components to alter its presentation mode, but clearly the user can explicitly request only one such change at a time. Following each user action, the system changes the presentation based on the user action, but also such that the overall presentation is the most preferred according to the document author's preferential constraints. Due to non-trivial preference constraints (modeled in this system by a CP-net[2]), each user action (a selection) may have multiple, non-trivial ramifications.

A key requirement for such systems is that the remote users browsing the information be provided with the quickest possible response time to their viewing choices. Ideally, we would have liked to download to the clients the whole document ahead of time. However, the buffer available on the client side may be much smaller than the size of the requested multimedia documents, and bandwidth limitations are likely to make downloading even a single document component upon request a lengthy process. Thus, we should strive to pre-fetch, during communication dead-time (i.e. during the time the user is looking at a component on the display, but performing no actions that require network communication), the components that are likely to be a part of the document presentation in the near future.

In [13] we present the corresponding pre-fetching mechanism for system similar to those discussed in [11, 12]. If a user-model (i.e., a distribution over future user selections) is available, one can use it to

2. CP-net is a graphical model for compactly representing preferences. Like Bayes networks, CP-nets consist of a directed acyclic graph, where each node represents a domain variable. Each node contains a conditional preference table, similar in size to a Bayes network CPT. However, the semantics of a CP-net is a partial preference ordering over outcomes, rather than a joint probability distribution. See [1, 2] for an introduction to CP-nets.

predict the distribution over the required future system actions. In such a system, we can model the likelihood of alternative document presentations by a probability distribution, compactly represented by a Bayes network over the document components. For the specific applications described above, the Bayes network is derived by copying the topology of the CP-net, replacing the conditional preferences with the appropriate conditional probabilities, and adding the selector node S as a parent of all the variables. Each variable corresponds to a component, and each variable state corresponds to a presentation mode for that component. Computing marginals in this network is thus equivalent to finding the marginal probability of the presentation mode for each component. This information can be used to determine what to deploy to the buffer on the client side, and what to remove (suppress) from this buffer. In the applications outlined above, prediction is actually achieved by computing marginals without evidence. However, we can certainly envision cases where the capability for handling evidence would be required.

Exploiting special properties of CP-nets in this mechanism results in a prediction scheme with time complexity of $O(n^2d)$, where n is the (typically large) number of information components, and d is the (typically small) maximum number of alternative presentation options for each component. The results of this paper are directly applicable there, reducing time complexity in certain cases to $O(nd^2)$. Due to the algorithmic focus of this paper, for the exact details of this specific prediction scheme we refer the reader to [13].

5.2 Related work

Some related work relevant to taking advantage of various belief network special-case characteristics is briefly discussed below. Our algorithm for belief updating in DP-singly connected BNs can be seen as an extension of the polytree belief updating algorithm [17]. In fact, some of the quantities computed in our algorithm are exactly π messages from [17], and similar quantities. Likewise, for the general evidence case, our extended algorithm uses both π and λ messages, as well as other types of messages that are specific to aggregated states of the selector S.

Alternately, one can view our algorithm within the framework of refined conditioning schemes presented in [8, 9], but where we take additional advantage of context specific independence and the selector properties of S. As written, the schemes proposed in [8, 9] would still have a quadratic runtime, even for networks that are polytrees (with one additional common selector). However, one may envision using a variant of recursive conditioning that conditions on partitions of S - that may be an alternate method of solving the problem in linear time, that may also be conducive to applying advantage of mutual exclusion in more general network topologies.

Yet another way to look at our algorithm is within the framework of Symbolic Probabilistic Inference (SPI [7]). Their system attempts to perform (automated) factoring of the symbolic equations for belief updating, and in theory could arrive at results similar to our scheme. However, this is unlikely in practice, without adding in rules for handling mutual exclusion and specific search heuristics - an interesting issue for future research.

Finally, a useful approach for exploiting context-specific independence in probabilistic inference is presented in [21]. This approach is based on the notion of contextual factors that encode dynamic direct dependencies between the variables — which variables act as parents may depend on the value of other variables. The representation and inference scheme in [21] is attractive in cases where the contextual factors associated with each variables are compact. Otherwise, at present, the standard methods seem to be computationally more efficient [21]. In the case of Bayes networks with selectors considered in our work, their algorithm can save by representing efficiently the dependency of each node on the selector. Specifically, once *partial* information is known about the selector state, many nodes in the network become contextually independent of the selector. However, [21] address the issue of multi-valued variables only partially, and certainly do not address mutual exclusion, a notion which is tangential to the notion of contextual independence. Thus, for the networks we are considering, their scheme (as it currently stands) would still run in quadratic time, whereas our

algorithm works in linear time. In future work, one could potentially apply the notion of mutual exclusion within the framework of [21] and possibly gain the advantages of both schemes.

5.3 Possible algorithm generalizations

We briefly discuss other possible future extensions to our algorithm. First, consider the case where there is more than one selector variable, in order to model more than one user selection (or other external event). If the topology is still DP-singly connected, except for the common selector variables, it seems that the above methods can still apply, but with a significantly larger number of cases to handle. For m selectors, this complexity should be nK^m for some small constant K. This is exponential in m, but complexity of existing algorithms will be $O(n^{m+1})$, so our scheme has potential here.

Relaxing the assumption that a selector affects only one variable at a time is also of interest. It should be possible to extend the scheme to some cases where the selector affects a small bounded number of nodes for each state of the selector, at the cost of a certain performance penalty. The conditions under which this would work are an issue for future research. Further possible extensions would be to consider dependent selectors, as well as other topologies. In a general topology, with one selector, one could apply a join-tree algorithm to the network (excluding S). Possibly, some form of mutual exclusion can be used for the resulting join-tree in the presence of S. Due to clustering effects, the mutual exclusion property may hold only partially, but it may still be possible to use it to gain some performance improvement. Finally, it may be possible to take advantage of forms of mutual exclusion other than selectors.

6. Summary

We examined a special case of Bayes network used in an application that requires prediction of component values, or system actions, in order to improve system performance in some criteria (such as average response time). One such application, adaptive multimedia presentation, was briefly discussed.

Such networks have a selector variable S as a parent of all other nodes. In particular, we examine cases where the network (excluding S) is singly connected. Belief updating in Bayes networks with this topology, can be done by conditioning on S in quadratic time, which is the best that existing algorithms can achieve. However, if the relationship between S and its children is such that S only affects at most node at a time, one can use the resulting mutual exclusion and context-specific independence - which we have done, resulting in a linear-time algorithm for belief updating in this topology.

Two variants of the algorithm were discussed. The first works for all networks that have directed-path singly connected topology (excluding the selector). For such networks, our algorithm works in linear time if there are no evidence nodes. The second variant works in networks with arbitrary conjunctive evidence, but in this case only for the more restricted case of polytree topology (again, excluding the selector).

Our work leaves several interesting avenues for future work, from generalizing to several selector variables, to other types of mutual exclusion. Taking advantage of mutual exclusion in networks that are multiply connected is also an interesting challenge.

Acknowledgments

This research was partially supported by an infrastructure grant from the Israeli Science Ministry, and by the Paul Ivanier Center for Robotics and Production Management, Ben-Gurion University. Carmel Domshlak was partially supported by the Intelligent Information Systems Institute, Cornell University (AFOSR grant F49620-01-1-0076). The views and conclusions contained herein are

those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government.

Appendix

The message equations are derived in this appendix, proving the theorems in Section 4. We begin with a context-specific independence lemma, used to argue independence throughout.

Lemma 1 *Let $\mathcal{S}$ be some knowledge about the selection, stating that S is known* not *to select a set of variables $\mathcal{X}$. Let $\mathcal{B}_{\mathcal{S}}$ be a Bayes network graph resulting from removing all arcs in $\mathcal{B}$ from node S to all variables in set $\mathcal{X}$. Let Z be an arbitrary set of nodes in $\mathcal{B}$, and X, Y be any two nodes in $\mathcal{B}$ that are not in Z. If X is d-separated from Y given Z in $\mathcal{B}_{\mathcal{S}}$, then X is independent of Y given Z and $\mathcal{S}$ in $\mathcal{B}$.*

Proof: The idea is that for cases where S is known *not* to select a set of variables $\mathcal{X}$, we can replace S by two nodes - call them $S_{\mathcal{X}}$ and $S_{\neg\mathcal{X}}$. The former, $S_{\mathcal{X}}$ is a new root node, that has as children all nodes in $\mathcal{X}$, as well as the node $S_{\neg\mathcal{X}}$. The node $S_{\neg\mathcal{X}}$ has as children all nodes except for $S_{\mathcal{X}}$ and the nodes in $\mathcal{X}$. Let $D(S_{\mathcal{X}}) = \bigcup_{X\in\mathcal{X}} D(X) \cup \perp$, and $D(S_{\neg\mathcal{X}}) = D(S) - D(S_{\mathcal{X}}) \cup \perp$. All children of the nodes $S_{\mathcal{X}}$, $S_{\neg\mathcal{X}}$ retain the same conditional probabilities. The distribution $P(S_{\neg\mathcal{X}}|S_{\mathcal{X}} = \perp)$ is equal to the posteriors over S given that S does not select any variable in $\mathcal{X}$. Conditional probabilities given other values of $S_{\mathcal{X}}$ can be set arbitrarily. (Note that the cardinality of $S_{\neg\mathcal{X}}$ may be smaller than that of S, but the values of S not represented in $S_{\neg\mathcal{X}}$ have a posterior probability of 0.) In the new network, add as evidence $S_{\mathcal{X}} = \perp$, denoting that no variable in $\mathcal{X}$ is selected. By construction, the distribution of the new network with this evidence is equal to that of the original network given that S is known *not* to select a set of variables $\mathcal{X}$. Thus, d-separation in the new network implies independence in the original network. Specifically, $S_{\mathcal{X}}$ is an evidence root node, and thus all undirected paths passing through this node are blocked. Thus, as far as testing d-separation is concerned, we can drop $S_{\mathcal{X}}$ and all its arcs from the network. The topology of the resulting network is the same as that of $\mathcal{B}_{\mathcal{S}}$, proving the lemma. □

We proceed with the derivation of equation 8, repeated below for convenience.

$$P(X|e_X^+) = P(X|S_X, e_X^+)P(S_X|e_X^+) + P(X|S_X^+, e_X^+)P(S_X^+|e_X^+) + P(X|S_X^{'-}, e_X^+)P(S_X^{'-}|e_X^+) =$$

$$\frac{P(S_{X=x})}{P(S_X)}\sigma_X + \pi_X^+\sigma_X^+ + \pi_X^0\sigma_X^{'-}$$

We claimed that $P(X|S_X^{'-}, e_X^+) = P(X|S_0) = \pi_X^0$ holds because X is independent of S given e_X^+ when S is known not to select a variable upstream from X. This is shown by using Lemma 1, as follows. Let $\mathcal{X}$ be the set all nodes upstream from X, as well as X itself. According to Lemma 1, we can remove all arcs from S to $\mathcal{X}$ before testing d-separation. In the modified network, every simple path from $\mathcal{X}$ to $S_{\neg\mathcal{X}}$ must pass only through nodes downstream from X, since the original network (without S) is a polytree. Since all evidence in e_X^+ is upstream from X, the above paths must all be blocked, and thus X is d-separated from $S_{\neg\mathcal{X}}$ given $S_X^{'-}, e_X^+$, and independence holds as required.

In the derivation of the first term, we have $P(X|S_X, e_X^+) = P(S_{X=x}|S_X, e_X^+)$, because once we know S_X, i.e. that S selects a value for X, the distribution of X follows the selection deterministically. Now, using an argument similar that used in the proof of Lemma 1 we can conclude that the selection is independent of e_X^+ given S_X, and thus:

$$P(X|S_X, e_X^+) = P(S_{X=x}|S_X) = \frac{P(S_{X=x}, S_X)}{P(S_X)} = \frac{P(S_{X=x})}{P(S_X)}$$

Putting it all together completes the derivation of equation 8, computing $P(X|e_X^+)$.

Proof of Theorem 2: Starting with $P(S'^{-}_X|e^+_X)$, we have, using Bayes:

$$P(S'^{-}_X|e^+_X) = \frac{P(e^+_X|S'^{-}_X)P(S'^{-}_X)}{P(e^+_X)} = \frac{P(e^+_X|S_0)P(S'^{-}_X)}{P(e^+_X)} = \frac{\pi^{e0}_X P(S'^{-}_X)}{P(e^+_X)}$$

where the second equality follows from Lemma 1 - independence given that we know that there is no selection upstream from X. Computing $P(S_X|e^+_X)$ is similar to the above:

$$P(S_X|e^+_X) = \frac{P(e^+_X|S_X)P(S_X)}{P(e^+_X)} = \frac{P(e^+_X|S_0)P(S_X)}{P(e^+_X)} = \frac{\pi^{e0}_X P(S_X)}{P(e^+_X)}$$

To compute $P(S^+_X|e^+_X)$ we sum over selection events:

$$P(S^+_X|e^+_X) = \sum_{U \in Pa(X)} P(S^{U+}_X|e^+_X)$$

and proceed to compute each term using Bayes equation:

$$P(S^{U+}_X|e^+_X) = \frac{P(e^+_X|S^{U+}_X)P(S^{U+}_X)}{P(e^+_X)}$$

The first numerator term decomposes (due to Lemma 1):

$$P(e^+_X|S^{U+}_X) = \prod_{W \in Pa(X)} P(e^{W+}_X|S^{U+}_X) = P(e^{U+}_X|S^{U+}_X) \prod_{W \in Pa(X)\setminus\{U\}} P(e^{W+}_X|S_0)$$

$$= \pi^{e+}_{U\to X} \prod_{W \in Pa(X)\setminus\{U\}} \pi^{e0}_{W\to X}$$

□

Proof of Theorem 3: Proceeding with derivation of equation 12, by conditioning on selection in the subtree anchored at each of $Pa(X)$, we have:

$$\pi^+_X = P(X|S^+_X, e^+_X) = \sum_{U \in Pa(X)} P(X|S^{U+}_X, e^+_X)P(S^{U+}_X|S^+_X, e^+_X) =$$

$$\sum_{U \in Pa(X)} P(X|S^{U+}_X, e^+_X)\frac{P(S^{U+}_X)P(e^+_X|S^{U+}_X)}{P(e^+_X)P(S^+_X|e^+_X)} = \sum_{U \in Pa(X)} P(X|S^{U+}_X, e^+_X)\frac{P(S^{U+}_X)}{P(e^+_X)\sigma^+_X} \prod_{W \in Pa(X)} P(e^{W+}_X|S^{U+}_X)$$

$$= \frac{1}{P(e^+_X)\sigma^+_X} \sum_{U \in Pa(X)} P(X|S^{U+}_X, e^+_X)P(S^{U+}_X)\pi^{e+}_{U\to X} \prod_{W \in Pa(X)\setminus\{U\}} \pi^{e0}_{W\to X}$$

For the first term in the summation above we have, by conditioning on assignments to parents of X:

$$P(X|S^{U+}_X, e^+_X) = \sum_{\mathbf{v} \in D(Pa(X))} P(X|\mathbf{v}, S^{U+}_X, e^+_X)P(\mathbf{v}|S^{U+}_X, e^+_X) = \sum_{\mathbf{v} \in D(Pa(X))} P(X|\mathbf{v}, S_0)P(\mathbf{v}|S^{U+}_X, e^+_X)$$

and where the last term can be further decomposed:

$$P(\mathbf{v}|S^{U+}_X, e^+_X) = P(\mathbf{v}(U)|S^{U+}_X, e^{U+}_X) \prod_{W \in Pa(X)\setminus\{U\}} P(\mathbf{v}(W)|S_0, e^{W+}_X)$$

$$= \pi^+_{U\to X}(\mathbf{v}(U)) \prod_{W\in Pa(X)\setminus\{U\}} \pi^0_{W\to X}(\mathbf{v}(W))$$

where $\pi^+_{U\to X}(\mathbf{v}(U))$ denotes the element of the (vector) message indexed by $\mathbf{v}(U)$. Substituting the above results into the preceding equations results in equation 12.
Now to derive equation 13. We have:

$$\pi^+_{U\to X} = P(U|S^{U+}_X, e^{U+}_X) = \frac{P(U, S^{U+}_X, e^{U+}_X)}{P(S^{U+}_X, e^{U+}_X)} = \frac{P(U, S^{U+}_X, e^{U+}_X)}{P(S^{U+}_X)P(e^{U+}_X|S^{U+}_X)} = \frac{P(U, S^{U+}_X, e^{U+}_X)}{P(S^{U+}_X)\pi^{e+}_{U\to X}}$$

The numerator becomes, by further partitioning S:

$$P(U, S^{U+}_X, e^{U+}_X) = P(U, S_U, e^{U+}_X) + P(U, S^+_U, e^{U+}_X) + \sum_{V\in Ch(U)\setminus\{X\}} P(U, S^{V-}_U, e^{U+}_X) \qquad (30)$$

Beginning with the first term in Eq. 30, we note that it is 0 whenever the value for U is not equal to the value selected in S, and that selecting a value for U in S results in U having that value with probability 1. Thus:

$$P(U = u, S_U, e^{U+}_X) = P(S_{U=u}, e^{U+}_X) = P(e^{U+}_X|S_{U=u})P(S_{U=u})$$

The first term becomes:

$$P(e^{U+}_X|S_{U=u}) = P(e^+_U|S_{U=u}) \prod_{W\in Ch(U)\setminus\{X\}} P(e^{W-}_U|U = u, S_0)$$

$$= \pi^{e0}_U \prod_{W\in Ch(U)\setminus\{X\}} \lambda^0_{W\to U}(u)$$

where the last equality follows from Lemma 1 and message definitions. Now, to the second term in Eq. 30:

$$P(U, S^+_U, e^{U+}_X) = P(S^+_U)P(U, e^{U+}_X|S^+_U) = P(S^+_U)P(U, e^+_U, \bigcup_{W\in Ch(U)\setminus\{X\}} e^{W-}_U|S^+_U)$$

$$= P(S^+_U)P(e^+_U|S^+_U)P(U|e^+_U, S^+_U) \prod_{W\in Ch(U)\setminus\{X\}} P(e^{W-}_U|U, S_0)$$

$$= P(S^+_U)\pi^{e+}_U\pi^+_U \prod_{W\in Ch(U)\setminus\{X\}} \lambda^0_{W\to U}$$

The elements of the summation in Eq. 30 are re-written:

$$P(U, S^{V-}_U, e^{U+}_X) = P(S^{V-}_U)P(U, e^{U+}_X|S^{V-}_U) = P(S^{V-}_U)P(U, e^+_U, \bigcup_{W\in Ch(U)\setminus\{X\}} e^{W-}_U|S^{V-}_U)$$

$$= P(S^{V-}_U)P(e^+_U|S_0)P(U|e^+_U, S_0)P(e^{V-}_U|U, S^{V-}_U) \prod_{W\in Ch(U)\setminus\{X,V\}} P(e^{W-}_U|U, S_0)$$

$$= P(S^{V-}_U)\pi^{e0}_U \quad \pi^0_U \quad \lambda^-_{V\to U} \prod_{W\in Ch(U)\setminus\{X,V\}} \lambda^0_{W\to U}$$

Substituting in all the above results and factoring out common factors, we get equation 13. □

Proof of Theorem 4: We begin with the derivation of $\pi^{e+}_{U\to X}$. By conditioning using a partition of S^U_X, we have:

$$\pi^{e+}_{U\to X} = P(e^{U+}_X|S^{U+}_X) = P(e^{U+}_X|S_U)P(S_U|S^{U+}_X) + P(e^{U+}_X|S^+_U)P(S^+_U|S^{U+}_X) + \tag{31}$$

$$+ \sum_{Y\in Ch(U)\setminus\{X\}} P(e^{U+}_X|S^{Y-}_U)P(S^{Y-}_U|S^{U+}_X)$$

Now develop each of the conditional evidence probability terms in Eq. 31, by conditioning on U:

$$P(e^{U+}_X|S_U) = \sum_{u\in D(U)} P(e^{U+}_X, U=u|S_U) =$$

$$= P(e^+_U|S_U) \sum_{u\in D(U)} P(U=u|e^+_U, S_U) \prod_{Y\in Ch(U)\setminus\{X\}} P(e^{Y-}_U|U=u, e^+_U, S_U)$$

$$= \frac{P(e^+_U|S_0)}{P(S_U)} \sum_{u\in D(U)} P(S_{U=u}) \prod_{Y\in Ch(U)\setminus\{X\}} P(e^{Y-}_U|U=u, S_0)$$

$$= \frac{\pi^{e0}_U}{P(S_U)} \sum_{u\in D(U)} P(S_{U=u}) \prod_{Y\in Ch(U)\setminus\{X\}} \lambda^0_{Y\to U}(u)$$

The second conditional evidence probability term in equation 31 becomes by conditioning:

$$P(e^{U+}_X|S^+_U) = \sum_{u\in D(U)} P(e^{U+}_X, U=u|S^+_U)$$

$$= P(e^+_U|S^+_U) \sum_{u\in D(U)} P(U=u|e^+_U, S^+_U) \prod_{Y\in Ch(U)\setminus\{X\}} P(e^{Y-}_U|U=u, e^+_U, S^+_U)$$

$$= P(e^+_U|S^+_U) \sum_{u\in D(U)} P(U=u|e^+_U, S^+_U) \prod_{Y\in Ch(U)\setminus\{X\}} P(e^{Y-}_U|U=u, S_0)$$

$$= \pi^{e+}_U \sum_{u\in D(U)} \pi^+_U(u) \prod_{Y\in Ch(U)\setminus\{X\}} \lambda^0_{Y\to U}(u)$$

We remain with the third conditional evidence probability term in equation 31 (inside the summation), which becomes:

$$P(e^{U+}_X|S^{Y-}_U) = \sum_{u\in D(U)} P(e^{U+}_X, U=u|S^{Y-}_U) =$$

$$= P(e^+_U|S^{Y-}_U) \sum_{u\in D(U)} P(U=u|e^+_U, S^{Y-}_U) P(e^{Y-}_U|S^{Y-}_U, U=u) \prod_{W\in Ch(U)\setminus\{X,Y\}} P(e^{W-}_U|S^{Y-}_U, U=u)$$

$$= P(e_U^+|S_0) \sum_{u \in D(U)} P(U=u|e_U^+, S_0) P(e_U^{Y-}|S_U^{Y-}, U=u) \prod_{W \in Ch(U) \setminus \{X,Y\}} P(e_U^{W-}|S_0, U=u)$$

$$= \pi_U^{e0} \sum_{u \in D(U)} \pi_U^0(u) \lambda_{Y \to U}^-(u) \prod_{W \in Ch(U) \setminus \{X,Y\}} \lambda_{W \to U}^0(u)$$

Putting it all together and factoring out common λ^0 terms, we get equation 16.
To derive $\pi_{U \to X}^{e0}$ we condition on U:

$$\pi_{U \to X}^{e0} = P(e_X^{U+}|S_0) = \sum_{u \in D(U)} P(e_X^{U+}, U=u|S_0)$$

$$= P(e_U^+|S_0) \sum_{u \in D(U)} P(U=u|e_U^+, S_0) P(\bigcup_{Y \in Ch(U) \setminus \{X\}} e_U^{Y-}|U=u, S_0))$$

$$= \pi_U^{e0} \sum_{u \in D(U)} \pi_U^0(u) \prod_{Y \in Ch(U) \setminus \{X\}} \lambda_{Y \to U}^0(u)$$

which results in equation 18.
□

Proof of Theorem 5: To derive $\lambda_{X \to U}^-$, we condition on selection events from the view-point of X:

$$\lambda_{X \to U}^- = P(e_U^{X-}|S_U^{X-}, U) = P(e_U^{X-}, S_0|S_U^{X-}, U) + P(e_U^{X-}, S_X|S_U^{X-}, U) + \tag{32}$$

$$+ P(e_U^{X-}, S_X^-|S_U^{X-}, U) + \sum_{Z \in Pa(X) \setminus \{U\}} P(e_U^{X-}, S_X^{Z+}|S_U^{X-}, U)$$

The first term in Eq. 32 is 0, since S_0 is incompatible with S_U^{X-}. For the second term in Eq. 32, we have:

$$P(e_U^{X-}, S_X|S_U^{X-}, U) = P(e_U^{X-}|S_X, U) P(S_X|S_U^{X-}, U) = P(e_U^{X-}|S_X, U) P(S_X|S_U^{X-}) = \frac{P(e_U^{X-}|S_X, U) P(S_X)}{P(S_U^{X-})}$$

where the the first part of the denominator requires more work. Following Pearl, we will denote by V the (temporarily constructed for convenience) composite variable consisting of $Pa(X) \setminus \{U\}$ - i.e. all parents of X excluding U.

$$P(e_U^{X-}|S_X, U) = \sum_{x \in D(X)} \sum_{\mathbf{v} \in D(V)} P(e_U^{X-}|\mathbf{v}, X=x, S_X, U) P(\mathbf{v}, X=x|S_X, U)$$

Now, the evidence above and below X is d-separated by X and S_X from the rest of the evidence, as well as from v and U. Likewise, evidence above V is separated by V and S_X from X, and from anything below X, and from U. Additionally, selection at X is the same as null selection as far as evidence probability is concerned. Thus:

$$P(e_U^{X-}|S_X, U) = \sum_{x \in D(X)} \sum_{v \in D(V)} P(e_X^-|X=x, S_0) P(e_X^{V+}|v, S_0) P(v, X=x|S_X, U)$$

Now, using Bayes rule, we get:

$$P(e_U^{X-}|S_X,U) = \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_X^-|X=x,S_0)\frac{P(e_X^{V+}|S_0)P(v|e_X^{V+},S_0)}{P(v|S_0)}P(X=x|v,S_X,U)P(v|U,S_0)$$

$$= \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_X^-|X=x,S_0)\frac{P(e_X^{V+}|S_0)P(v|e_X^{V+},S_0)}{P(v|S_0)}P(X=x|v,S_X,U)P(v|S_0)$$

$$= P(e_X^{V+}|S_0)\sum_{x\in D(X)} P(S_{X=x}|S_X)P(e_X^-|X=x,S_0)\sum_{v\in D(V)} P(v|e_X^{V+},S_0)$$

$$= P(e_X^{V+}|S_0)\sum_{x\in D(X)} \frac{P(S_{X=x})}{P(S_X)}\lambda_X^0(x)$$

where:

$$P(e_X^{V+}|S_0) = \prod_{Z\in Pa(X)\setminus\{U\}} P(e_X^{Z+}|S_0) = \prod_{Z\in Pa(X)\setminus\{U\}} \pi_{Z\to X}^{e0}$$

For the third term in equation 32, we have:

$$P(e_U^{X-},S_X^-|S_U^{X-},U) = P(e_U^{X-}|S_X^-,U)P(S_X^-|S_U^{X-},U) = P(e_U^{X-}|S_X^-,U)P(S_X^-|S_U^{X-})$$

Since the last term above is a simple computation over priors of S, we need only consider the first term - similar to the above method - with V as before:

$$P(e_U^{X-}|S_X^-,U) = \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_U^{X-}|v,X=x,S_X^-,U)P(v,X=x|S_X^-,U)$$

Using d-separation arguments as above, we have:

$$P(e_U^{X-}|S_X^-,U) = \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_X^-|X=x,S_X^-)P(e_X^{V+}|v,S_0)P(v,X=x|S_0,U)$$

Now, using Bayes rule, we get:

$$P(e_U^{X-}|S_X^-,U) = \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_X^-|X=x,S_X^-)\frac{P(e_X^{V+}|S_0)P(v|e_X^{V+},S_0)}{P(v|S_0)}P(X=x|v,S_0,U)P(v|S_0) =$$

$$P(e_X^{V+}|S_0)\sum_{x\in D(X)} P(e_X^-|X=x,S_X^-)\sum_{v\in D(V)}\prod_{Z\in Pa(X)\setminus\{U\}} M_X(x|v,U)P(v(Z)|e_X^{Z+},S_0) =$$

$$P(e_X^{V+}|S_0)\sum_{x\in D(X)} \lambda_X^-(X=x)\sum_{v\in D(V)} M_X(x|v,U)\prod_{Z\in Pa(X)\setminus\{U\}} \pi_{Z\to X}^0(v(Z))$$

In the summation in Eq. 32, evaluating the terms, we have (for Z being an arbitrary parent of X, excluding U):

$$P(e_U^{X-},S_X^{Z+}|S_U^{X-},U) = P(e_U^{X-}|S_X^{Z+},U)P(S_X^{Z+}|S_U^{X-})$$

Again, the last term is a simple computation on priors of S, and the first term we solve by conditioning (again, V is the parent composite variable):

$$P(e_U^{X-}|S_X^{Z+},U) = \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_U^{X-}|v,X=x,S_X^{Z+},U)P(v,X=x|S_X^{Z+},U)$$

Using d-separation arguments as above, we have:

$$P(e_U^{X-}|S_X^{Z+},U) = \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_X^-|X=x,S_0)P(e_X^{V+}|v,S_X^{Z+})P(v,X=x|S_X^{Z+},U)$$

Now, using Bayes rule, we get:

$$P(e_U^{X-}|S_X^{Z+},U) = \sum_{x\in D(X)} \sum_{v\in D(V)} P(e_X^-|X=x,S_0)\frac{P(e_X^{V+}|S_X^{Z+})P(v|e_X^{V+},S_X^{Z+})}{P(v|S_X^{Z+})}P(X=x|v,S_0,U)P(v|S_X^{Z+}) =$$

$$P(e_X^{V+}|S_X^{Z+}) \sum_{x\in D(X)} \lambda_X^0(X=x) \sum_{v\in D(V)} M_X(x|v,U)P(v|e_X^{V+},S_X^{Z+})$$

Now, separating out the nodes in the composite variable V into its components, W, we have:

$$P(v|e_X^{V+},S_X^{Z+}) = P(v(Z)|e_X^{Z+},S_X^{Z+})) \prod_{W\in Pa(X)\setminus\{U,Z\}} P(v(W)|e_X^{W+},S_0) =$$

$$\pi_{Z\to X}^+(v(Z)) \prod_{W\in Pa(X)\setminus\{U,Z\}} \pi_{W\to X}^0(v(W))$$

and likewise:

$$P(e_X^{V+}|S_X^{Z+}) = P(e_X^{Z+}|S_X^{Z+}) \prod_{W\in Pa(X)\setminus\{U,Z\}} P(e_X^{W+}|S_0)$$

$$= \pi_{Z\to X}^{e+} \prod_{W\in Pa(X)\setminus\{U,Z\}} \pi_{W\to X}^{e0}$$

Putting it all together, we get equation 20 □

Proof of Theorem 6: For convenience, we will develop the equation for the message λ_U^- instead. By conditioning over selection of subtrees anchored at X, we have:

$$\lambda_U^- = P(e_U^-|S_U^-,U) = \sum_{X\in Ch(U)} P(e_U^-|S_U^{X-},U)P(S_U^{X-}|S_U^-,U) = \sum_{X\in Ch(U)} P(e_U^-|S_U^{X-},U)\frac{P(S_U^{X-})}{P(S_U^-)}$$

Due to d-separation of the partial evidence given U and the information about the selector, and that again, once we know there is no selection downstream from the subtree anchored at W, it is equivalent to having state S_0, we have:

$$P(e_U^-|S_U^{X-},U) = P(e_U^{X-}|S_U^{X-},U) \prod_{W\in Ch(U)\setminus\{X\}} P(e_U^{W-}|S_0,U)$$

$$= \lambda^{-}_{X \to U} \prod_{W \in Ch(U) \setminus \{X\}} \lambda^{0}_{W \to U}$$

Substituting the last equation into the previous one, and re-writing everything in terms of X rather than U, we get equation 21. □

To conclude, we must show that the equations are well defined. We use the convention that a sum over zero terms is zero, and that a product over zero terms is 1. First, all σ_X message types require only π messages from parents of X (or no messages, if X is a root node), and priors over the selector - and are thus well defined.

Next, we need to show that all the π and λ messages are well defined. In general, since the definition pattern follows the scheme from the Kim and Pearl algorithm [20], the set of equations we use are likewise well defined. Specifically, examine the π vector messages. π^{+}_{X} uses a σ message and $P(e^{+}_{X})$, which only need the value of messages from parents of X. It also requires other types of π messages, but always from parents of X. Message $\pi^{+}_{U \to X}$ requires λ messages, but only from nodes *other* than X. The same definition pattern obviously holds for the π^{0}_{X} and $\pi^{0}_{U \to X}$ messages. In general, the pattern is as follows: generating a message always uses only messages that are closer to the external nodes of the polytree. Messages for the most external nodes are the base cases, and are always well defined. The only exceptions to these rules occur when we use some expressions as shorthand. For example, in equation 16 which defines $\pi^{e+}_{U \to X}$, we have the term $\frac{\lambda^{0}_{X}(u)}{\lambda^{0}_{X \to U}(u)}$ which uses a λ message from X to U. However, this is just a convenient shorthand, since

$$\frac{\lambda^{0}_{X}(u)}{\lambda^{0}_{X \to U}(u)} = \prod_{W \in Ch(X) \setminus \{U\}} \lambda^{0}_{W \to U}(u)$$

References

[1] C. Boutilier, R. Brafman, C. Domshlak, H. Hoos, and D. Poole. CP-nets: A tool for representing and reasoning about conditional *ceteris paribus* preference statements. *Journal of Artificial Intelligence Research*, 2004. to appear.

[2] C. Boutilier, R. Brafman, H. Hoos, and D. Poole. Reasoning with Conditional Ceteris Paribus Preference Statements. In *Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence*, pages 71–80. Morgan Kaufmann Publishers, 1999.

[3] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Uncertainty in Artificial Intelligence, Proceedings of the 12th Conference*, pages 115–123. Morgan Kaufmann, August 1996.

[4] E. Charniak. Bayesian networks without tears. *AI Magazine*, 12(4):50–63, Winter 1991.

[5] G. F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42(2-3):393–405, 1990.

[6] P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60(1):141–153, 1993.

[7] B. D'Ambrosio. Incremental probabilistic inference. In *Uncertainty in AI, Proceedings of the 9th Conference*, July 1993.

[8] A. Darwiche. Conditioning methods for exact and approximate inference in causal networks. In *Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence*, pages 99–107, 1995.

[9] A. Darwiche. Recursive conditioning. *Artificial Intelligence*, 125(1-2):5–41, 2000. Special Issue on Resource Bounded Reasoning.

[10] F. J. Diez. Local conditioning in bayesian networks. *Artificial Intelligence*, 87(1-2):1–20, 1996.

[11] C. Domshlak, R. Brafman, and S. E. Shimony. Preference-based Configuration of Web Page Content. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 1451–1456, Seattle, August 2001.

[12] C. Domshlak, R. Brafman, and S. E. Shimony. Qualitative decision making in adaptive presentation of structured information. *ACM Transactions on Information Systems (TOIS)*, 2004. to appear.

[13] C. Domshlak and S. E. Shimony. Improving the dynamic behavior of CP-net based multimedia systems by predicting likely components. In *AAAI/KDD/UAI Joint Workshop on Real-Time Decision Support and Diagnosis Systems*, Edmonton, Canada, July 2002. `http://www.cs.cornell.edu/~dcarmel/Papers/Source/prefetch.rtdsds02.ps.gz`.

[14] E. Gudes, C. Domshlak, and N. Orlov. Remote Conferencing with Multimedia Objects. In *Proceedings of the Second International Workshop on Multimedia Data Document Engineering (MDDE'02)*, March 2002.

[15] E. J. Horvitz, H. J. Suermondt, and G. F. Cooper. Bounded conditioning: Flexible inference for decisions under scarce resources. In *5th Workshop on Uncertainty in AI*, August 1989.

[16] F. V. Jensen, K. G. Olsen, and S. K. Andersen. An algebra of Bayesian belief universes for knowledge-based systems. *Networks*, 20:637–660, 1990.

[17] J. H. Kim and J. Pearl. A Computation Model for Causal and Diagnostic Reasoning in Inference Systems. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, 1983.

[18] S.L. Lauritzen and David J. Speigelhalter. Local computations with probabilities on graphical structures and their applications to expert systems. *Journal of the Royal Statistical Society*, 50:157–224, 1988.

[19] R. E. Neapolitan. *Probabilistic Reasoning in Expert Systems*. John Wiley and Sons, 1990.

[20] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[21] D. Poole and N. L. Zhang. Exploiting contextual independence in probabilistic inference. *Journal of Artificial Intelligence Research*, 18:263–313, 2003.

[22] S. Russel and P. Norvig, editors. *Artificial Intelligence - A Modern Approach*. Prentice Hall, 1995.

[23] S. E. Shimony and C. Domshlak. Complexity of probabilistic reasoning in directed-path singly connected Bayes networks. *Artificial Intelligence*, 151(1-2):213–225, 2003.

[24] Solomon E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, August 1994.