Many-Pairs Mutual Information for Adding Structure to Belief Propagation Approximations

Arthur Choi and **Adnan Darwiche**
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90095
{*aychoi,darwiche*}@*cs.ucla.edu*

Abstract

We consider the problem of computing mutual information between many pairs of variables in a Bayesian network. This task is relevant to a new class of Generalized Belief Propagation (GBP) algorithms that characterizes Iterative Belief Propagation (IBP) as a polytree approximation found by deleting edges in a Bayesian network. By computing, in the simplified network, the mutual information between variables across a deleted edge, we can estimate the impact that recovering the edge might have on the approximation. Unfortunately, it is computationally impractical to compute such scores for networks over many variables having large state spaces. So that edge recovery can scale to such networks, we propose in this paper an approximation of mutual information which is based on a soft extension of d-separation (a graphical test of independence in Bayesian networks). We focus primarily on polytree networks, which are sufficient for the application we consider, although we discuss potential extensions of the approximation to general networks as well. Empirically, we show that our proposal is often as effective as mutual information for the task of edge recovery, with orders of magnitude savings in computation time in larger networks. Our results lead to a concrete realization of GBP, admitting improvements to IBP approximations with only a modest amount of computational effort.

Introduction

Iterative belief propagation (IBP), often referred also to as loopy belief propagation (Pearl 1988; Yedidia, Freeman, and Weiss 2005), is an approximate inference algorithm that has been critical for enabling certain classes of applications which could never be handled using exact algorithms due to the high treewidth of their associated models. For example, Turbo decoding, then described as "the most exciting and potentially important development in coding theory in many years", as well as low density parity check decoding, have since been shown to be instances of belief propagation in Bayesian networks with loops (Frey and MacKay 1997; McEliece, MacKay, and Cheng 1998). IBP has also spawned approaches such as Survey Propagation, which have shown to be capable of solving particularly difficult instances of the satisfiability problem (Braunstein, Mézard, and Zecchina 2005). IBP has also been applied towards a variety of computer vision tasks (Szeliski et al. 2006), particularly stereo vision, where methods based on belief propagation have been particularly successful.

The successes of IBP as an approximate inference algorithm spawned a number of generalizations, in the hopes of understanding the nature of IBP, and further to improve on the quality of its approximations. Among the most notable are the family of Generalized Belief Propagation (GBP) algorithms (Yedidia, Freeman, and Weiss 2005), where the "structure" of an approximation is given by an auxiliary graph called a region graph, which specifies (roughly) which regions of the original network should be handled exactly, and to what extent different regions should be mutually consistent. IBP occurs as an instance of GBP when the regions and mutual consistencies are those implied by polytrees. In order to improve on the quality of an IBP approximation, one must specify the appropriate structure (i.e., the region graph) used to drive the GBP algorithm. This by itself can be a nontrivial task, as networks may not lend themselves to an obvious choice of regions.

More recently, we identified a special class of GBP approximations called ED-BP (Choi and Darwiche 2006), which characterized IBP as an algorithm in a polytree approximation of the original network, found by deleting enough edges; cf. (Wainwright, Jaakkola, and Willsky 2003). Edge deletion gives rise to a spectrum of approximations, with IBP on one end (with a polytree approximation) and exact inference on the other (when every edge is recovered). In order to identify good approximations within this spectrum, we can take advantage of the polytree approximation, using it to find dependencies that call for the *recovery* of edges. To this end, we proposed a mutual information heuristic for edge recovery that we demonstrated to be effective in improving IBP approximations.

Unfortunately, when many edges are deleted, we must compute the mutual information between many pairs of variables, which can be computationally impractical even in polytrees. The problem becomes increasingly worse for networks over many variables, more so when variables have large state spaces. Therefore, a more efficient approach to the edge recovery task, and more generally, to the task of choosing the structure for BP approximations, can potentially extend the success of IBP and GBP algorithms to applications beyond what they currently reach.

We propose here an approximation to the mutual information that is based on a soft extension of the *d-separation* tests used for identifying conditional independencies in Bayesian networks (Pearl 1988). As d-separation verifies the blockage of information flow between variables of interest, our soft alternative estimates the degree to which this information flow is throttled. We focus on the key setting for edge recovery, polytree networks, where our soft extension provides an upper bound on the mutual information between a given pair of variables. We explicate the issues that arise for the problem of many-pairs mutual information, and illustrate the utility of our alternative in the edge recovery task. In particular, we find empirically that our soft d-separation tests effectively approximate the *ranking* of edges based on mutual information, which in this and many other applications, is the true task at hand. We further discuss potential extensions of soft d-separation to general networks as well.

Edge Deletion and Belief Propagation

In (Choi and Darwiche 2006), we proposed an approach to approximate inference that is based on performing exact inference in a Bayesian network that has been simplified by deleting edges, where the deletion of an edge further introduces free parameters into the approximation. We further characterized a set of parameters for deleted edges that induced a class of approximations called ED-BP that:

- yield exact marginals when a single edge deletion splits a network into two independent subnetworks;
- yield iterative belief propagation (IBP) as a special case of edge deletion when the approximate network is reduced to a polytree (any polytree), where IBP approximations for node and family marginals are the exact node and family marginals in the polytree ED-BP approximation $\mathcal{N}'$.
- implied an iterative algorithm (also called ED-BP) for parametrizing the approximation, in a manner analogous to message passing in IBP.

When $\mathcal{N}'$ is multiply-connected (i.e., has undirected cycles), then ED-BP induces a class of GBP approximations.[1] Therefore, by choosing edges to delete, we implicitly choose also the structure of a GBP approximation.

By deleting edges in a model, we can simplify it until it is amenable to exact inference. In turn, we can exploit the simplified model and search for dependencies that call for the *recovery* of edges. Indeed, in (Choi and Darwiche 2006), we proposed a heuristic that ranked deleted edges based on mutual information scores computed in the polytree approximation. By measuring the mutual information between the variables across a deleted edge,[2] they try to estimate the extent to which an edge's deletion split the network into two

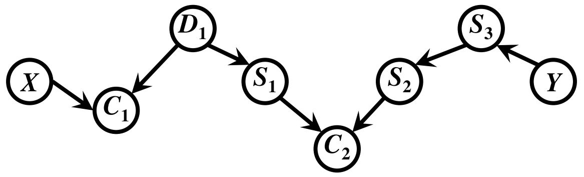

Figure 1: A path with 6 valves. Sequential valves are labeled S_i, divergent valves are labeled D_j and convergent valves are labeled C_k.

(the exact case). As the splitting case implies a mutual information of zero, edges with the highest mutual information were chosen for recovery first. Their empirical results suggest that this heuristic is indeed effective in finding good edges to recover. We find in practice, however, that this approach is often computationally impractical, for example, in large networks where many edges are deleted, for which the mutual information must be computed for many pairs.

d-Separation: A Brief Review

The d-separation test for Bayesian networks is a graphical test for determining whether or not two variables X and Y are conditionally independent given another set of variables $\mathbf{Z}$. Although d-separation can identify any independence implied by the structure of a Bayesian network, it cannot in general detect an independence implied by its parameters.

Intuitively, X and Y are d-separated by $\mathbf{Z}$ when every path connecting X and Y are blocked by variables in $\mathbf{Z}$, where blocked is a notion we define below. When this d-separation exists, we denote it by $\text{d-sep}(X, \mathbf{Z}, Y)$, and conclude that X and Y are conditionally independent given $\mathbf{Z}$.

Consider then a path connecting X to Y, that need not be a directed path, such as the one in Figure 1. Say we view this path as a *pipe*, and view each variable W on that path as a *valve*. A valve is either *open* or *closed*, and if at least one valve in a pipe is closed, then the enter pipe is blocked; otherwise, it is unblocked. When all possible pipes connecting X and Y are blocked, then we say that X and Y are d-separated.

A node can be one of three different valves on a given path; see, e.g., Figure 1. A node W in a path is either (1) a *sequential* valve $\rightarrow W \rightarrow$, (2) a *divergent* valve $\leftarrow W \rightarrow$, or (3) a *convergent* valve $\rightarrow W \leftarrow$, depending on the directionality of the incident edges as it appears on the path. A valve is then open or closed depending on the set of variables $\mathbf{Z}$.

- A sequential valve $\rightarrow W \rightarrow$ or a divergent valve $\leftarrow W \rightarrow$ is considered closed when W appears in $\mathbf{Z}$.
- A convergent valve $\rightarrow W \leftarrow$ is considered closed when neither W nor any of its descendants appear in $\mathbf{Z}$.

In Figure 1, for example, variables X and Y are d-separated given no variables $\mathbf{Z}$, as both convergent valves are closed. When $\mathbf{Z}$ contains both of these valves, X and Y are no longer d-separated. When we further add any of the remaining nodes to $\mathbf{Z}$, variables X and Y are d-separated again.

Note that when we need to refer to a valve's neighbors, we will refer to a valve, for example, by $N_1 \rightarrow W \leftarrow N_2$.

[1]An ED-BP approximation $\mathcal{N}'$ corresponds to an instance of GBP run with a particular joingraph (Choi and Darwiche 2006; Aji and McEliece 2001; Dechter, Kask, and Mateescu 2002).

[2]When an edge $U \rightarrow X$ is deleted, the arc is removed, and a clone $\hat{U}$ replaces U as a parent of X. The score given to each deleted edge was then basically $MI(U; \hat{U} \mid \mathbf{e})$.

Soft d-Separation: A Proposal for Polytrees

Consider a *polytree* Bayesian network $\mathcal{N}$, with nodes X and Y. Further, say we are given $\mathbf{z}$, an instantiation of some network variables $\mathbf{Z}$. If we have that $\textsf{d-sep}(X, \mathbf{Z}, Y)$, then X is independent of Y given $\mathbf{Z}$. Even if this is not the case, variables X and Y may still be independent, or only mildly so.

We can quantify the extent to which two variables are independent using mutual information:

$$MI(X;Y|\mathbf{Z}=\mathbf{z}) \overset{def}{=} \sum_{xy} Pr(xy|\mathbf{z}) \log \frac{Pr(xy|\mathbf{z})}{Pr(x|\mathbf{z})Pr(y|\mathbf{z})}.$$

Note that mutual information is non-negative, and zero iff X and Y are independent given $\mathbf{z}$. Consider also the entropy:

$$ENT(X \mid \mathbf{Z}=\mathbf{z}) \overset{def}{=} \sum_{x} Pr(x \mid \mathbf{z}) \log \frac{1}{Pr(x \mid \mathbf{z})}.$$

Entropy can be thought of as a measure of uncertainty. It is non-negative, and zero iff X is fixed. Entropy is also maximized by a uniform distribution. For more on mutual information and entropy, see (Cover and Thomas 1991). Note that the above are defined for distributions $Pr(.|\mathbf{Z}=\mathbf{z})$ conditioned on a given instantiation $\mathbf{Z}=\mathbf{z}$ (or just simply $\mathbf{z}$).

Since we are assuming for now that $\mathcal{N}$ is a polytree, there is a unique path connecting X and Y (assuming that they are connected). Consider first a node W that is either a sequential valve $\rightarrow W \rightarrow$ or a divergent valve $\leftarrow W \rightarrow$. When W is in $\mathbf{Z}$, the valve is closed, and X and Y are d-separated in the polytree. When $W \notin \mathbf{Z}$, we now have a way to think of a valve as being *partially* closed.

Theorem 1 *In a polytree Bayesian network $\mathcal{N}$, if a node W is a sequential valve $\rightarrow W \rightarrow$ or a divergent valve $\leftarrow W \rightarrow$ on a path connecting X and Y, then:*

$$MI(X;Y \mid \mathbf{z}) \leq ENT(W \mid \mathbf{z}).$$

When the entropy of W is zero, then W is given, effectively closing the valve. We have also the following, as a way to think of convergent valves as partially closed.

Theorem 2 *In a polytree Bayesian network $\mathcal{N}$, if a node W is a convergent valve $N_1 \rightarrow W \leftarrow N_2$ on a path connecting X and Y, then:*

$$MI(X;Y \mid \mathbf{z}) \leq MI(N_1;N_2 \mid \mathbf{z}).$$

Note that when $MI(N_1;N_2|\mathbf{z}) = 0$, no information is flowing through the valve, as N_1 and N_2 in effect do not "see" the instantiation $\mathbf{z}$ at or below its common descendant W.

Note in the latter case, we bound a mutual information by another mutual information. The upper bound in this case involves the mutual information between two variables, N_1 and N_2, that are members of the same family[3] (that of variable W). This distinction is important in practice as BP and jointree algorithms readily provide any family's marginals. We discuss this issue further in the following section.

We are now prepared to propose our soft d-separation test in polytrees. Intuitively, we are examining the pipe connecting X and Y, searching for the tightest valve, as the tightest valve can be considered the bottleneck over the entire pipe.

[3]A family consists of a node and its parents.

Definition 1 *Let X and Y be two nodes in a polytree Bayesian network $\mathcal{N}$, conditioned on instantiation $\mathbf{z}$. Then the value of $\textsf{sd-sep}(X,\mathbf{z},Y)$ is defined by three cases:*

- *X and Y are disconnected: $\textsf{sd-sep}(X,\mathbf{z},Y) = 0$;*
- *X and Y are adjacent: $\textsf{sd-sep}(X,\mathbf{z},Y) = MI(X;Y|\mathbf{z})$;*
- *otherwise: let $\textsf{sd-sep}(X,\mathbf{z},Y)$ be the smallest upper bound of a valve W on the path connecting X and Y, as given by Theorems 1 and 2.*

Note if X and Y are adjacent, they are in the same family.

Corollary 1 *If X and Y are two nodes in a polytree Bayesian network $\mathcal{N}$, conditioned on instantiation $\mathbf{z}$, then:*

$$MI(X;Y \mid \mathbf{z}) \leq \textsf{sd-sep}(X,\mathbf{z},Y).$$

Since $\textsf{sd-sep}(X,\mathbf{z},Y)$ is either zero, an entropy, or a mutual information, it is both finite and non-negative. Note that when $\textsf{d-sep}(X,\mathbf{Z},Y)$, we also have $\textsf{sd-sep}(X,\mathbf{z},Y) = 0$, as expected. Soft d-separation, on the other hand, can detect more conditional independencies than is allowed by examining the network structure alone.

We can potentially tighten this bound, by incorporating $ENT(X|\mathbf{z})$ and $ENT(Y|\mathbf{z})$, as both quantities are also upper bounds on the mutual information $MI(X;Y|\mathbf{z})$. We shall employ this simple extension in our experiments.

Many Pairs Mutual Information

The edge recovery heuristic of (Choi and Darwiche 2006) scores each deleted edge based on the mutual information, computed in the polytree approximation, between the variables across the deleted edge. When many edges are deleted, as may be needed to obtain a polytree from a given network, we need to compute the mutual information for many pairs of variables. Although exact inference in a polytree network is generally tractable, computing mutual information for many pairs can still be impractical.

Consider the fact that Pearl's belief propagation (BP) algorithm (Pearl 1988) can compute in a single run the marginals for all variables, and the joint marginals for all families. Since a score $\textsf{sd-sep}(X,\mathbf{z},Y)$ is either an entropy (requiring only a variable's marginals) or a mutual information between two family members (requiring only a family's marginals), we can compute with a single run of BP the score $\textsf{sd-sep}(X,\mathbf{z},Y)$ for any given X, Y and $\mathbf{z}$. In fact, we can compute with a single run of BP *all* valve bounds (of Theorems 1 and 2) for a given instantiation $\mathbf{z}$. By annotating the network graph with valve bounds in a pre-processing step, we can in fact compute the score $\textsf{sd-sep}(X,\mathbf{z},Y)$ for any given pair (X,Y) simply by inspecting the graph, much like the original d-separation algorithm.

Say we have a polytree Bayesian network with n variables, where each variable has at most w parents and at most s states. Then Pearl's BP algorithm has a worst-case running time $O(ns^w)$ (the largest CPT in a given network is of size $O(s^w)$). Given a graph annotated with valve bounds, it takes $O(n)$ time to compute $\textsf{sd-sep}(X,\mathbf{z},Y)$, so we can compute soft d-separation scores for k pairs in time $O(kn + ns^w)$.

Consider then the computation of a mutual information $MI(X;Y|\mathbf{z})$, when X and Y are not in the same family. We need the table of probabilities $Pr(X, Y|\mathbf{z})$, which (naively) requires us to run BP $O(|X| \cdot |Y|) = O(s^2)$ times; employing a simple technique, we can reduce this to $O(s)$ runs of BP.[4] This by itself can be undesirable, when the number of states s is very large. When we *further* want to compute the mutual information for many pairs (X, Y), the number of times we need to run BP can easily grow out of hand.

To compute the mutual information between k pairs of variables, we may need to run BP in a polytree as many as $O(ks)$ times, giving us a time complexity of $O(ks \cdot ns^w)$ total. In contrast, soft d-separation for k pairs required only $O(kn + ns^w)$ time. For a given application we may be able to identify improvements (we do so for mutual information, in the following section), but many runs of BP will still be undesirable in practice. Finally, note that edge recovery, our application of interest, starts with an ED-BP approximation that was computed using the exact BP algorithm as a subroutine, i.e., when we are ready to compute soft d-separation scores, valve bounds have in effect already been computed.

Experimental Evaluation

We evaluate now the *ranking* of variable pairs for the purposes of edge recovery, where deleted edges are ranked by soft d-separation (SD-SEP) and by mutual information (TRUE-MI). In particular, we:

1. find a (random) spanning polytree approximation $\mathcal{N}'$, by deleting enough edges in $\mathcal{N}$;
2. parametrize deleted edges in $\mathcal{N}'$ using ED-BP, (equivalently, run the IBP algorithm in $\mathcal{N}$);
3. score deleted edges by SD-SEP and TRUE-MI, which we compute in the polytree approximation $\mathcal{N}'$;
4. recover the highest ranking edges into $\mathcal{N}'$, by SD-SEP and TRUE-MI, parametrizing the new networks using ED-BP.

Our primary concern is the efficiency of Step 3, and the quality of the approximation resulting in Step 4. We typically expect that recovery of edges with high TRUE-MI scores in Step 3 to benefit most the ED-BP approximation in Step 4.

In our experiments, we took a selection of Bayesian networks,[5] comparing the edge scores produced by SD-SEP and TRUE-MI, and the resulting ED-BP approximations. Consider Figure 2, where each column corresponds to a particular Bayesian network, which is evaluated based on an average of 100 problem instances. Each problem instance corresponds to observations $\mathbf{e}$ on all leaves of the original network, whose values are sampled from the original joint distribution (for network `emdec`, we set values on leaves at random as the joint distribution is highly skewed). We dropped instances where ED-BP failed to converge, in Step 2, to within a difference of 10^{-8}, in 200 iterations.

In the first two rows of Figure 2, we compared the edge scores given by TRUE-MI and the bound SD-SEP. On the x-axis are edge ranks for the TRUE-MI score: the highest ranking edges are on the left, and the lowest on the right; on the y-axis are edge scores. The TRUE-MI scores decrease monotonically as we move right on the x-axis, and ideally, the SD-SEP scores should decrease monotonically as well. In the first row of plots, a small fraction of edges dominate the rest, in terms of both TRUE-MI and SD-SEP scores. In the second row, which plots the same scores under a logarithmic scale, the SD-SEP upper bound tends to be tighter in the range of high ranking edges. Moreover, at high values of TRUE-MI, the bounds are well behaved in that edges with higher TRUE-MI scores on average tend to have higher SD-SEP scores as well. We consider this range to be the most important for accurate ranking, as recovering these edges are likely to benefit the final ED-BP approximation the most. In the range of low TRUE-MI, the SD-SEP scores are relatively smaller, but can be poorly behaved. This is not surprising however, since at this range TRUE-MI scores are zero or close to zero, and are pairwise indistinguishable.

On the third row of Figure 2, we plot how edge scores compare when recovering edges, which is ultimately what we care about the most. We recovered edges into a polytree, $\frac{1}{10}$-th of the deleted edges at a time, until all edges are recovered. From left-to-right, edges with the largest scores were recovered first, from the case corresponding to IBP (when no edge is recovered) to the case corresponding to exact inference (when every edge is recovered). We also plot for reference approximations based on random edge recovery.

In these networks, edge recovery based on SD-SEP and TRUE-MI rankings are very much in step. In the `emdec` network, recovery by SD-SEP seems modestly better than TRUE-MI. As TRUE-MI is a heuristic, and non-optimal, this can be expected. Another interesting trend we observe is that edge recovery is most effective for edges with relatively high scores. When recovering the remaining edges, approximations improve as modestly as random recovery, suggesting that adaptive heuristics, where edges are re-ranked after recovery, should be considered when recovering more edges.

In Table 3, we compare in larger networks: the quality of the resulting approximation, and the time to compute rankings.[6] Remember that scores are computed in the polytree approximation, with each deleted edge receiving a score. To compute the TRUE-MI scores, we used a Hugin (jointree) variant of the BP algorithm to compute joint marginals, making an effort to reduce the number of propagations required.[7] For computing the SD-SEP scores, we measured the time to inspect the graph and assign scores to pairs, as described in the previous section. We see that for most networks, ranking by SD-SEP is as good (or better) than TRUE-MI. Moreover, SD-SEP is quite efficient, especially for larger networks with many parameters (implying roughly an increased cost for a run of BP), and with more edges deleted (implying more runs of BP). In large networks, SD-SEP can be faster by multiple orders of magnitude. For example, in the `diabetes` network, SD-SEP offers a $641.99\times$ speedup over TRUE-MI.

[4] We need only $O(\min\{|X|, |Y|\}+1) = O(s)$ runs of BP since $Pr(X, Y|\mathbf{z}) = Pr(X|Y, \mathbf{z})Pr(Y|\mathbf{z}) = Pr(Y|X, \mathbf{z})Pr(X|\mathbf{z})$.

[5] Most of the networks used for our evaluation are available at http://www.cs.huji.ac.il/labs/compbio/Repository/. Network `emdec` is a noisy-or network for diagnosis, courtesy of HRL Laboratories, LLC.

[6] Measured on a 2.16GHz Intel Core 2 Duo with 1GB RAM.

[7] If there are T number of tails U of deleted edges, then just $O(T \cdot \max_U |U|)$ propagations suffice (Choi and Darwiche 2006).

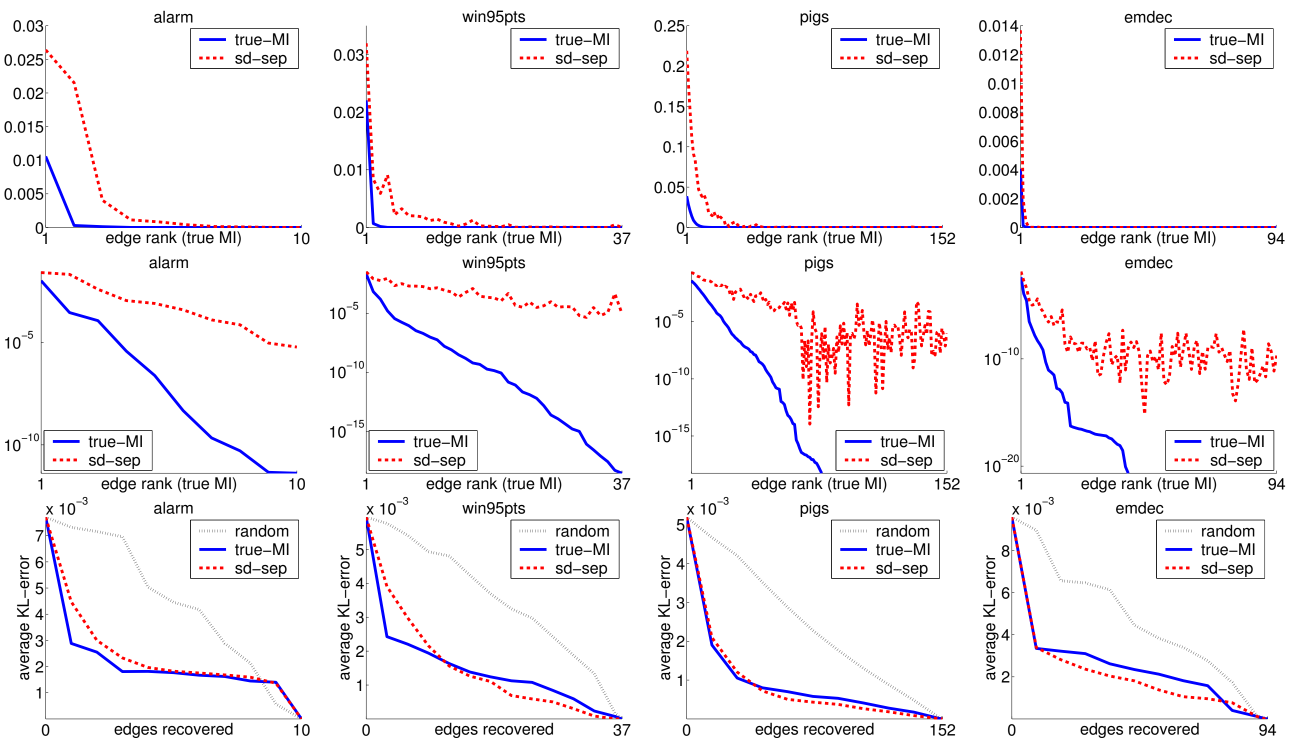

Figure 2: (Quality of SD-SEP approximations) Each column corresponds to a Bayesian network. First row (linear scale) and second row (log scale) compares edge rank by TRUE-MI versus edge scores by SD-SEP and TRUE-MI. Third row compares the number of edges recovered versus the quality of the ED-BP approximation (by KL-error of marginals).

	edge	0% recovered			10% recovered			20% recovered			edge	# of	# of
	recovery method	avg KL error	ED-BP time	(iterations)	avg KL error	ED-BP time	(iterations)	avg KL error	ED-BP time	(iterations)	ranking time	edges deleted	network params.
`barley`	random	1.82e-1	115ms	(65)	1.70e-1	120ms	(61)	1.67e-1	141ms	(58)	0ms	37	130180
	MI				1.62e-1	111ms	(59)	1.41e-1	93ms	(49)	2999ms		
	sd-sep				1.65e-1	110ms	(60)	1.62e-1	125ms	(55)	46ms	**65.84**×	
`diabetes`	random	3.52e-2	732ms	(123)	3.24e-2	1103ms	(109)	3.02e-2	1651ms	(93)	0ms	190	461069
	MI				2.61e-2	550ms	(74)	2.12e-2	674ms	(58)	84604ms		
	sd-sep				2.93e-2	957ms	(74)	2.54e-2	1639ms	(55)	132ms	**641.99**×	
`mildew`	random	1.12e-2	238ms	(39)	1.10e-2	241ms	(38)	1.06e-2	243ms	(35)	0ms	12	547158
	MI				7.68e-3	233ms	(36)	7.26e-3	263ms	(33)	6661ms		
	sd-sep				8.28e-3	245ms	(35)	7.01e-3	323ms	(31)	42ms	**157.26**×	
`munin1`	random	2.84e-4	13ms	(23)	2.89e-4	14ms	(21)	2.84e-4	22ms	(19)	0ms	94	19466
	MI				2.83e-4	12ms	(22)	2.22e-4	10ms	(19)	680ms		
	sd-sep				2.58e-4	10ms	(20)	2.21e-4	10ms	(18)	35ms	**19.57**×	
`munin2`	random	2.44e-4	54ms	(22)	2.48e-4	51ms	(21)	2.30e-4	52ms	(18)	0ms	242	83920
	MI				9.44e-5	37ms	(17)	8.46e-5	37ms	(16)	8294ms		
	sd-sep				1.05e-4	39ms	(17)	7.80e-5	36ms	(16)	286ms	**29.03**×	
`munin3`	random	2.77e-5	32ms	(13)	2.75e-5	36ms	(12)	2.78e-5	54ms	(11)	0ms	272	85855
	MI				2.56e-5	30ms	(11)	2.47e-5	39ms	(10)	8675ms		
	sd-sep				2.25e-5	33ms	(12)	1.77e-5	30ms	(11)	375ms	**23.14**×	
`munin4`	random	4.71e-4	67ms	(27)	4.75e-4	78ms	(28)	4.61e-4	89ms	(23)	0ms	357	98183
	MI				3.56e-4	64ms	(23)	3.45e-4	50ms	(18)	12277ms		
	sd-sep				3.37e-4	54ms	(20)	3.35e-4	58ms	(19)	518ms	**23.69**×	

Figure 3: (Comparison of recovery heuristics). For $\{0, 10, 20\}\%$ of the edges recovered, we report average KL-error of ED-BP marginal approximations and average running time (and # of iterations) for ED-BP to converge. For each recovery method, we report average time spent to rank edges, with relative improvement of SD-SEP over TRUE-MI annotated adjacent to its time.

Alternative Proposals and Extensions

Ideally, a true extension of d-separation would be applicable to general networks and not just polytrees. Roughly, we want to aggregate the scores for each pipe, and identify an approximation to the mutual information $MI(X;Y|\mathbf{z})$. To this end, we propose the following procedure. First, identify a maximal set of pairwise node-disjoint paths connecting X and Y (disjoint in the interior nodes). For each of the node-disjoint paths, we compute the score as if the corresponding pipe were in a tree, as in Definition 1. The score $\mathsf{sd\text{-}sep}(X,\mathbf{z},Y)$ now, is then the *sum* of these individual pipe scores. Intuitively, we are treating the set of tightest valves over all node-disjoint pipes as the bottlenecks to flow from X to Y. Although, $\mathsf{sd\text{-}sep}(X,\mathbf{z},Y)$ is an upper bound on the mutual information in a polytree, this particular extension sacrifices this property in general networks. We consider these issues and extensions ongoing research.

Note that to compute node-disjoint paths, we can use Dinic's algorithm (Even 1979), having time complexity $O(n^{\frac{1}{2}}m)$, where n is the number of nodes and m is the number of edges in a Bayesian network $\mathcal{N}$. We can also use the Ford-Fulkerson algorithm which, in this case, has time complexity $O(w(n+m))$, where w is the treewidth of $\mathcal{N}$; see, e.g., (Kleinberg and Tardos 2005). Assuming we use a jointree algorithm, we again need just one jointree propagation to score all valves. To compute the soft d-separation score of each desired pair, we incur a time complexity that is only polynomial in the parameters of the graph. Computing the mutual information for many pairs, on the other hand, may require many jointree propagations, each requiring time that is exponential in the treewidth w of the graph.

Alternatively, we could consider a proposal such as the one given by (Nicholson and Jitnah 1998), which suggests a heuristic approach to quantifying the relevance of paths, also using mutual information. In particular, the weight of a path was the sum of the mutual informations between variables on an edge, for each edge on the path. While this proposal could conceivably be used to approximate the mutual information between two nodes, we find that an additive accumulation of scores on a path unsuitable. For example, an additive score may give high weight to two nodes connected by a path with many edges, yet a single vacuous edge on that path would render the two nodes independent.

Acknowledgments

This work has been partially supported by Air Force grant #FA9550-05-1-0075 and by NSF grant #IIS-0713166.

Proofs

Theorems 1 and 2 follow easily from the proof of the data processing inequality in (Cover and Thomas 1991). For completeness, we provide the proofs here.

Let W be a valve in a path connecting X and Y. Using the chain rule for mutual information, we can express the mutual information $MI(X;Y,W|\mathbf{z})$ in two different ways.

$$\begin{aligned} MI(X;Y,W \mid \mathbf{z}) &= MI(X;Y \mid W,\mathbf{z}) + MI(X;W \mid \mathbf{z}) \\ &= MI(X;W \mid Y,\mathbf{z}) + MI(X;Y \mid \mathbf{z}). \end{aligned}$$

Let W be either a sequential valve $\rightarrow W \leftarrow$, or a divergent valve $\leftarrow W \rightarrow$. Then $W,\mathbf{Z}$ d-separates X and Y in the polytree, and $MI(X;Y \mid W,\mathbf{z}) = 0$. Further, since mutual information is non-negative:

$$\begin{aligned} MI(X;W \mid \mathbf{z}) &= MI(X;W \mid Y,\mathbf{z}) + MI(X;Y \mid \mathbf{z}) \\ &\geq MI(X;Y \mid \mathbf{z}). \end{aligned}$$

Since $ENT(W|\mathbf{z}) \geq MI(X;W|\mathbf{z})$, we have Theorem 1. Suppose instead that W is a convergent valve $N_1 \rightarrow W \leftarrow N_2$, assuming for now that $N_1 \neq X$ and $N_2 \neq Y$. On the path connecting X and Y, we know that N_2 must be either a sequential valve or a convergent valve, and that $N_2,\mathbf{Z}$ d-separates X and Y. Using the same reasoning as before, with N_2 in the place of W, we have that

$$MI(X;N_2 \mid \mathbf{z}) \geq MI(X;Y \mid \mathbf{z}).$$

Similarly, $N_1,\mathbf{Z}$ must d-separate X and N_2, so

$$MI(N_1;N_2 \mid \mathbf{z}) \geq MI(X;N_2 \mid \mathbf{z}).$$

If $N_1 = X$ or $N_2 = Y$, we simply skip the appropriate steps, giving us finally Theorem 2. □

References

Aji, S. M., and McEliece, R. J. 2001. The generalized distributive law and free energy minimization. In *Proc. 39th Allerton Conf. Commun., Contr., Computing*, 672–681.

Braunstein, A.; Mézard, M.; and Zecchina, R. 2005. Survey propagation: An algorithm for satisfiability. *Random Struct. Algorithms* 27(2):201–226.

Choi, A., and Darwiche, A. 2006. An edge deletion semantics for belief propagation and its practical impact on approximation quality. In *AAAI*, 1107–1114.

Cover, T. M., and Thomas, J. A. 1991. *Elements of information theory*. Wiley-Interscience.

Dechter, R.; Kask, K.; and Mateescu, R. 2002. Iterative join-graph propagation. In *UAI*, 128–136.

Even, S. 1979. *Graph Algorithms*. Computer Science Press.

Frey, B. J., and MacKay, D. J. C. 1997. A revolution: Belief propagation in graphs with cycles. In *NIPS*, 479–485.

Kleinberg, J., and Tardos, E. 2005. *Algorithm Design*. Addison-Wesley.

McEliece, R. J.; MacKay, D. J. C.; and Cheng, J.-F. 1998. Turbo decoding as an instance of Pearl's "belief propagation" algorithm. *IEEE J. Sel. Areas Commun.* 16(2):140–152.

Nicholson, A. E., and Jitnah, N. 1998. Using mutual information to determine relevance in Bayesian networks. In *PRICAI*, 399–410.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc., San Mateo, California.

Szeliski, R.; Zabih, R.; Scharstein, D.; Veksler, O.; Kolmogorov, V.; Agarwala, A.; Tappen, M. F.; and Rother, C. 2006. A comparative study of energy minimization methods for Markov random fields. In *ECCV (2)*, 16–29.

Wainwright, M. J.; Jaakkola, T.; and Willsky, A. S. 2003. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Trans. Inf. Theory* 49(5):1120–1146.

Yedidia, J.; Freeman, W.; and Weiss, Y. 2005. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Inf. Theory* 51(7):2282–2312.