Elimination is Not Enough: Non-Minimal Triangulations for Graphical Models

Chris D. Bartels *and* **Jeff A. Bilmes**
{bartels,bilmes}@ee.washington.edu
Department of Electrical Engineering
University of Washington
Seattle, WA 98195

UWEE Technical Report
Number UWEETR-2004-0010
June 2004

Department of Electrical Engineering
University of Washington
Box 352500
Seattle, Washington 98195-2500
PHN: (206) 543-2150
FAX: (206) 543-3842
URL: http://www.ee.washington.edu

Elimination is Not Enough: Non-Minimal Triangulations for Graphical Models

Chris D. Bartels and **Jeff A. Bilmes**
{bartels,bilmes}@ee.washington.edu
Department of Electrical Engineering
University of Washington
Seattle, WA 98195

University of Washington, Dept. of EE, UWEETR-2004-0010
June 2004

Abstract

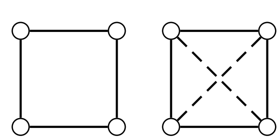
As is well known, exact probabilistic graphical inference requires a triangulated graph. Different triangulations can make exponential differences in complexity, but since finding the optimum is intractable, a wide variety of heuristics have been proposed, most involving a vertex elimination ordering. Elimination always yields a triangulated graph, can produce all edge minimal triangulations, and can also produce triangulations having the smallest maximal clique size. In this paper, we show that there are cases of practical importance where the optimal triangulation is unobtainable with elimination. Specifically, we show that real-world models with deterministic dependencies exist where the best elimination-based triangulation can have an unboundedly larger state space than the optimal triangulation. We present new methods that, unlike elimination, can generate optimal state space triangulations for such models, and give results for both real and randomly generated graphs. We also give an algorithm and correctness proof for determining if a triangulation can be obtained via elimination, and give a new proof that the decision problem associated with finding optimal state space triangulations is NP-complete.

1 INTRODUCTION

Exact inference in probabilistic graphical models requires a triangulated graph. There are many different triangulation choices each yielding exponential differences in inference time and memory complexity. Unfortunately, finding optimal triangulations is NP-hard [2, 25], so heuristic search methods must be used.

Indeed, many search methods exist, and are often based on choosing an ordering of the nodes for vertex elimination [8, 13, 14, 16, 19, 22]. Vertex elimination always results in a triangulated graph [22], but can not generate all possible triangulations (Figure 1). It can, however, create any minimal triangulation [20].

There are many methods to judge the quality of a given triangulation. Most basic is the size of the largest maximal clique of the graph. This is a good indicator on positive distributions where each random variable has the same cardinality. When the variables have different cardinalities, one typically looks at the state space of the junction tree [18, 4, 14, 16, 19, 25]. The optimal triangulation under these two criteria will always be minimal, and hence standard



(a) (b)

Figure 1: Triangulation which is impossible using elimination

techniques that choose an elimination order are perfectly sufficient. Typically, triangulation methods that are not based on elimination will also favor minimal triangulations and smaller clique sizes [6, 21].

Such is not the case, however, when one considers non-positive distributions. In many applications (such as hierarchical HMMs, speech recognition graphs, causal models, coding graphs, representations of context specific independence, etc.), many random variables might be deterministic functions of other random variables [10, 5, 7, 8, 9, 15], implying that combinations of values of certain variables will have zero probability. When these variables exist together in a maximal clique, significant reductions in state space can occur. A maximal clique, however, might not always contain the parents of a deterministic random variable necessary to achieve this state space reduction. This prompts the use of *non-minimal* triangulations which might better cluster such deterministically related variables. Such a triangulation, however, might not be obtainable through elimination on the original graph. We note in passing that even when the distributions are strictly positive, elimination might not be enough, as there are methods that enlarge cliques to reduce clique degree [1].

In this paper, we show that models with deterministic dependencies exist where the best elimination-based triangulation can have an unboundedly larger state space than the optimal triangulation. We present new methods that, unlike elimination, can generate optimal state space triangulations for such models, and we give results for both real and randomly generated graphs. We also give an algorithm and correctness proof for determining if a triangulation can be obtained via elimination, and we present complexity results for finding minimum state space triangulations.

Section 2 defines notation, necessary terminology, and re-visits important theorems from the literature. Section 3 discusses when elimination is a sufficient triangulation method and when it is not, and it relates elimination to marginalization in probabilistic models. Section 4 provides an algorithm for detecting when a given triangulated graph is obtainable via elimination starting from some un-triangulated graph. Section 5 discusses the complexity of finding an optimal triangulation under an arbitrary triangulation quality measure. Section 6 defines new triangulation methods that can find optimal triangulations in many cases where standard triangulation techniques will always fail. Lastly, Section 7 presents results on both real-world and random graphs, and Section 8 concludes. This paper is an expanded version of [3].

2 BACKGROUND

We first define notation, terminology, and mention a number of known theorems required in later sections. A **chord** is an edge connecting two non-consecutive vertices in a cycle of length greater than three. A graph is **triangulated** if it contains no chordless cycles. A **triangulation** of a graph $G = (V, E)$ is a (possibly empty) set of edges F such that $E \cap F = \emptyset$ and the graph $T(G) = (V, E \cup F)$ is triangulated. The edges in F are called **fill-in edges**. The term triangulation will also be used to mean the graph $T(G)$. Given a graph $G = (V, E)$, the **neighbors** of $v \in V$ are defined as $NE_G(v) = (w \in V | \{v, w\} \in E)$

Vertex elimination [22] is an algorithm that can be used to triangulate graphs. The algorithm chooses a vertex, adds edges so that all of its neighbors form a complete set, removes the vertex and its edges from the graph, and repeats until no vertices remain. One creates a triangulated graph by adding the edges that were added at each elimination step to the original graph. The order that the vertices are removed is the **elimination order**. An elimination order is a bijection $\alpha : \{1, 2, ..., |V|\} \leftrightarrow V$. We use $\alpha(v)$ to denote the integer position of node v in the ordering α, and $\alpha^{-1}(i)$ to denote the vertex indexed by the integer i in ordering α.

Definition 1. *The* **deficiency** *[22] of a vertex v in G is:* $D_G(v) = (\{u, w\} | \{v, u\} \in E, \{v, w\} \in E, \{u, w\} \notin E\})$

The **monotone deficiency** [22] of a vertex v in G is: $MD_G(v) = (D_G(v) \setminus \{u, w\} \mid \alpha^{-1}(u) < \alpha^{-1}(v))$. Given $G = (V, E)$, the v**-elimination** graph G_v is defined by adding the edges $D_G(v)$ and then deleting v and its incident edges from G. Creating G_v from G is known as eliminating the vertex v. The **elimination graph**, notated by $\mathcal{E}_\alpha(G)$, is the original graph G with the addition of any edges added at each step in the elimination process [22, p.600], or more formally:

$$\mathcal{E}_\alpha(G) = \left(V, E \bigcup_{i=1}^{|V|} MD(\alpha^{-1}(i)) \right)$$

A **perfect ordering** is an ordering for which elimination adds no edges, i.e., α is perfect if $\mathcal{E}_\alpha(G) = G$. A triangulation F of graph $G = (V, E)$ is **minimal** if $G' = (V, E \cup F_0)$ is not triangulated for any $F_0 \subset F$. A **non-minimal edge** of triangulation $T(G) = (V, E \cup F)$ is some $f \in F$ such that $G' = (V, E \cup F - \{f\})$ is triangulated. A vertex is

simplicial in the graph G if $NE_G(v)$ form a complete set. Note that the elimination of a simplicial vertex does not add any edges to the graph. A **clique** is a set of vertices for which every vertex in the set is connected to every other vertex in the set. A **maximal clique** is a clique that is not a subset of some larger clique.

Theorem 2. *All elimination graphs are triangulated. [22]*

Theorem 3. *A graph is triangulated if and only if it has a perfect elimination ordering. [22, Theorem 1].*

Lemma 4. *Triangulated graphs with more than one node have at least two simplicial vertices [12].*

Theorem 5. *Given a graph $G = (V, E)$ with triangulation F, F is a minimal triangulation if and only if for each $f \in F$, $G' = (V, E \cup F - \{f\})$ is not triangulated. [23, Theorem 1]*

Theorem 6. *Let $G = (V, E)$ be a graph and α be an elimination ordering. Then $\{v, w\}$ is a fill-in edge of $\mathcal{E}_\alpha$ if and only if there exists a chain $[v, v_1, v_2, ..., v_k, w]$ in G such that $\alpha(v_i) < \min(\alpha(v), \alpha(w))\, \forall\, i = 1...k$ [23, Lemma 4].*

Lemma 7. *Let $G = (V, E)$ be a triangulated graph and let $G' = (V, E')$ be a spanning triangulated subgraph of G with $|E \setminus E'| = k$. Then there is an increasing sequence $G' = G_0 \subset ... \subset G_k = G$ of triangulated graphs that differ by exactly one edge. [17, Lemma 2.21, page 20]*

Lemma 8. *Let $G = (V, E)$ be a triangulated graph and let $G' = (V, E')$ be a spanning triangulated subgraph of G with $|E \setminus E'| = k$. Then there is a decreasing sequence $G = G_0 \supset ... \supset G_k = G'$ of triangulated graphs that differ by exactly one edge.*

Proof. The sequence is the reverse of the sequence from G' to G as defined by Lemma 7. □

Although all elimination graphs are triangulated, and all triangulated graphs have perfect elimination orders, a lesser known fact is that some triangulations of a graph can not be generated via *any* elimination order ([20] and Figure 1). One important class, however, that can always be obtained via elimination is the minimal graphs.

Lemma 9. *Define $E(G)$ to be the edges of a graph G. Now consider a graph $G_1 = (V, E_1)$ which is a spanning subgraph of $G_2 = (V, E_2)$, i.e., $E_1 \subseteq E_2$. Then $E(\mathcal{E}_\alpha(G_1)) \subseteq E(\mathcal{E}_\alpha(G_2))$ and $E(\mathcal{E}_\alpha(G_1)) \setminus E(G_1) \subseteq E(\mathcal{E}_\alpha(G_2)) \setminus E(G_2)$ for any elimination order α.*

Proof. The lemma is obviously true for one vertex. Now assume it is true for $N - 1$ vertices and consider a graph with N vertices. $\text{NE}_{G_1}(\alpha^{-1}(1)) \subseteq \text{NE}_{G_2}(\alpha^{-1}(1))$, so $D_{G_1}(\alpha^{-1}(1)) \subseteq \Big(D_{G_2}(\alpha^{-1}(1)) \cup \big(E(G_2) \setminus E(G_1)\big)\Big)$. The $\alpha^{-1}(1)$-elimination graph $(G_1)_{\alpha^{-1}(1)}$ is therefore spanning subgraph of the $\alpha^{-1}(1)$-elimination graph $(G_2)_{\alpha^{-1}(1)}$ and the lemma is proven by the induction hypothesis. □

Theorem 10. *If $T(G) = (V, E \cup F)$ is a minimal triangulation of $G = (V, E)$, then there exists an elimination order α such that $\mathcal{E}_\alpha(G) = T(G)$ [20, Theorem 1].*

Proof. Choose α to be any perfect elimination order of $T(G)$, and eliminate G according to α creating $\mathcal{E}_\alpha(G) = (V, E \cup F_\alpha)$. G is a spanning subgraph of $T(G)$ so by Lemma 9 $F_\alpha \subseteq F$. By the definition of minimal there is no $F_0 \subset F$ such that $G = (V, E \cup F_0)$ is triangulated, so F_α must equal F. □

Corollary 11. *Given $T(G) = (V, E \cup F)$ is a minimal triangulation of $G = (V, E)$, if α is any perfect ordering of $T(G)$, $\mathcal{E}_\alpha(G) = T(G)$.*

Proof. Follows directly from the proof of Theorem 10. □

There are different methods to judge the quality of a triangulation. Two are defined here. First, the **treewidth** of a graph [2] is simply the size of its largest maximal clique minus 1. Second, the **state space** of a graph [18, 4, 14, 16, 19, 25] is widely used to analyze computational complexity of probabilistic inference. In [14], for example, various triangulation methods were analyzed using **weight** (the logarithm of the state space). Specifically, the **state space** or **cardinality** of a vertex v, notated $|v|$, is the number of distinct values its corresponding random variable may hold. The state space of a *clique*, C, holding vertices $v_1, v_2, ..., v_k$ is defined as $\text{S}(C) = \prod_{i=1}^{k} |v_i|$. Lastly, the state space of a graph, G, with maximal cliques $C_1, C_2, ..., C_k$ is defined as $\text{S}(G) = \sum_{i=1}^{k} S(C_i)$.

Next, we consider the use of deterministic variables inside the junction tree algorithm. When a deterministic variable is included in a clique with its parents, the probabilities of the variable never need to be stored in the clique potential because they will be known to be 1 or 0. When a message is passed to such a clique, the parents of the variable are iterated in outer loops, and the value of the deterministic variable is uniquely calculated. Hence, it does not contribute to memory usage and contributes only a constant factor in its parent's state space to the computation time. If the variable is in a maximal clique that is missing a parent, its memory and computation requirements are just as if it were random. An example of this in the context of marginalization will be given in Section 3.3. This leads us to modify our definition of clique state space when deterministic dependencies are present.

Definition 12. *The state space of a clique, C, with vertices $v_1, v_2, ..., v_k$, and with* $\mathcal{D} = \{v | v \text{ is deterministic and } \text{pa}(v) \in C\}$ *is defined as* $S(C) = \prod_{v \in V \setminus \mathcal{D}} |v|$.

3 ELIMINATION

Elimination is the most prominent algorithm used in triangulation searches and this section is devoted its discussion. First, it is proven that elimination is sufficient to find optimal treewidth triangulations and optimal state space triangulations when there are no deterministic relationships and no variables with cardinality 1. Note that observed random variables are computationally equivalent to having a cardinality of 1, so they are considered as such. Next, examples are given where elimination is not capable of finding an optimal triangulation. Last, marginalization of a variable is analogous to elimination of the variable on the graph. It is demonstrated that this analogy does not hold unaltered when a deterministic variable is present and when the optimal triangulation is desired.

3.1 WHEN ELIMINATION IS ENOUGH

There are many situations where optimal or nearly optimal triangulations may be achieved via elimination. If we consider the alternative, searching over *all* possible choices of fill-in, elimination has many desirable properties. For example, although there are $|V|!$ possible elimination orderings, there are $2^f = O(2^{|V|^2})$ possible fill-in choices (where f is the number of missing edges in the graph). More importantly, all elimination orders correspond to a triangulated graph but clearly arbitrary choices of fill-in do not. When the optimal triangulation, say $T(G)$, is minimal, the elimination graphs of G using *any* perfect ordering of $T(G)$ will yield $T(G)$. This implies that not only will elimination be sufficient, but also there will be quite a few such orders (at least $2^{|V|-1}$) that work (this follows from Lemmas 11, 4, and 3). Next it will be shown that elimination is sufficient for finding optimal treewidth triangulations and optimal state space triangulations for graphs having both positive distributions and hidden only nodes.

Theorem 13. *For any graph G some elimination graph of G will have optimal treewidth.*

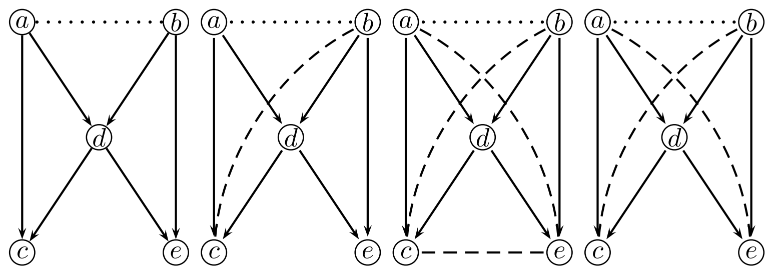
Proof. If we begin with some non-minimal optimal treewidth triangulation we can remove edges until some minimal triangulation is reached. When a non-minimal edge is removed it will either leave the treewidth unchanged or will lower it. This implies that some minimal triangulation will also have optimal treewidth, and from Theorem 10 this minimal triangulation will be an elimination graph. □

Lemma 14. *Consider a graph $G = (V, E)$ with any two maximal cliques C_1 and C_2. There exists nodes w_1 and w_2 such that $w_1 \in C_1$, $w_2 \in C_2$, $w_1 \notin C_2$, $w_2 \notin C_1$, and $(w_1, w_2) \notin E$.*

Proof. Suppose it is not true, then all $w_1 \in C_1 \setminus C_2$ are connected to all $w_2 \in C_2 \setminus C_1$. Because C_2 is maximal, there can not exist $v \in V \setminus C_2$ such that $\forall\ w_2 \in C_2$, $(w_2, v) \in E$. This is a contradiction because $C_1 \setminus C_2 \subseteq V \setminus C_2$ and $C_2 \setminus C_1 \subseteq C_2$. A symmetric argument holds starting with maximal C_1. □

Lemma 15. *Consider a triangulated graph $T(G) = (V, E \cup F)$ where an edge $(u, v) \in F$ exists such that $T'(G) = (V, E \cup (F \setminus (u, v)))$ is also triangulated. There is only one maximal clique in $T(G)$ containing both u and v.*

Proof. Assume the contrary, so there is more than one maximal clique in $T(G)$ containing u and v. We choose maximal cliques C_1 and C_2 both containing u and v, and choose vertices w_1 and w_2 such that $w_1 \in C_1$, $w_2 \in C_2$, $w_1 \notin C_2$, $w_2 \notin C_1$, and $(w_1, w_2) \notin E \cup F$. There will be a cycle in $T'(G)$ $u - w_1 - v - w_2 - u$, and because $T'(G)$ is triangulated this cycle must have a chord. Nodes w_1 and w_2 are not connected so u and v must be connected, but this is a contradiction because $T'(G)$ does not contain edge (u, v). □



(a) c, a, b, d (b) a, c, b, d (c) d, c, a, b (d) elimination not possible

Figure 2: Non-elimination based triangulation with unbounded improvement over elimination

Lemma 16. *Consider a triangulated graph $T(G) = (V, E \cup F)$ where there exists some edge (u, v) such that $T'(G) = (V, E \cup (F \setminus (u, v)))$ is also triangulated. Say that C is the* unique *maximal clique in $T(G)$ which contains both u and v. In $T'(G)$, C will be split into the cliques $C_u = C \setminus \{u\}$ and $C_v = C \setminus \{v\}$. We define c as the state space of the clique $C \setminus \{u, v\}$. If C_u and C_v are maximal cliques in $T'(G)$ the difference in state space between $T'(G)$ and $T(G)$ will be $c(|u| + |v| - |u||v|)$. If C_u is a subset of another maximal clique in $T'(G)$ the difference will be $(1 - |u|)c|v|$, if C_v is a subset it will be $(1 - |v|)c|u|$, and if both are subsets it will be $-c|u||v|$.*

Proof. The cliques not containing both u and v will be unaffected by the edge removal. The state space of C is $c|u||v|$, and the state spaces of C_v and C_u (if they exist) are $c|v|$ and $c|u|$ respectively. □

Lemma 17. *Given a graph $G = (V, E)$ where all variables are random (i.e., not deterministic) with state space ≥ 2, a minimal optimal state space triangulation will exist.*

Proof. Suppose we have a non-minimal triangulation $T(G)$ and remove an edge (u, v) creating new triangulation $T'(G)$. If either of the new cliques created by the edge removal are not maximal in $T'(G)$ the difference in state space will be negative since $|u| \geq 2$ and $|v| \geq 2$. When both cliques are maximal, the state space difference is $c(|u| + |v| - |u||v|)$, again negative. From Lemma 8, we can create a decreasing sequence of graphs from any non-minimal triangulation to some minimal triangulation, and the state space of each graph in the sequence will be less than or equal to the previous graph. Hence, any non-minimal triangulation will have a minimal spanning subgraph with a smaller or equal state space. □

Theorem 18. *Given a graph $G = (V, E)$ where all of the variables are random and have state space ≥ 2, some elimination graph of G will have optimal state space.*

Proof. By Lemma 17, a minimal optimal state space triangulation exists that by Lemma 10 is achievable using elimination. □

3.2 ELIMINATION IS NOT ENOUGH

When the value of at least one variable is a deterministic function of other variables, Theorem 18 no longer applies. Consider Figure 2 where d is deterministic in its parents a and b, the cardinalities of a, b, c and e are all η, and the cardinality of d is $\eta^2 - 1$ (the largest sensible cardinality for d). The state space of the graph with no additional fill-in edges is $2\eta^4 - \eta^2$. If one considers the graph in Figure 2(b) the cost is reduced to $\eta^4 + \eta^3 - \eta^2$. One might also run elimination beginning with d, resulting in the graph of Figure 2(c) and cost η^4. None of these nor any elimination ordering will give the optimal triangulation seen in Figure 2(d) having state space of $2\eta^3$ which is a factor of η faster than *any* elimination based triangulation. Although non-minimal triangulations can be useful in many contexts, finding triangulations for graphs with deterministic variables is the main focus of this work. One might also notice that in this example, the problem can be solved by transforming the graph into one which does not include the deterministic variable, and standard elimination can be used on the transformed graph. Although this approach could work, it will

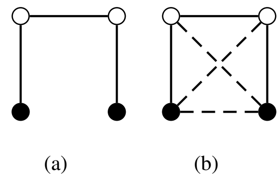


Figure 3: Optimal state space triangulation with observations

be shown in Sections 6.1.3 and 7.1.1 that the optimal choice of which transformations to make is not any simpler than choosing a fill-in. Keeping the triangulation search as a choice of fill-in is a much more uniform, elegant, and simple solution than a combining a search over both graph transformations and minimal fill-in (again, see Section 7.1.1).

When some variables are cardinality 1 (e.g., by treating observed variables computationally as having cardinality 1), the requirements for Theorem 18 are not met. For example, if one considers the graph in Figure 3(a) the optimal state space triangulation, given in Figure 3(b), is not obtainable through elimination. However, we can bound how poorly elimination can do.

Theorem 19. *We consider a graph $G = (V, E)$ where all of the variables are random with an arbitrary state space. $T(G) = (V, E \cup F)$ is an optimal state space triangulation with a state space of $\mathcal{O}$. Suppose there is some triangulated spanning subgraph $T'(G) = (V, E \cup F')$ with state space $\mathcal{E} > \mathcal{O}$. If $\mathcal{E}_{max}$ is the largest state space of any maximal clique in $T'(G)$ then $\mathcal{E} < \mathcal{O} + |V|^2 \mathcal{E}_{max}$.*

Proof. We can create a decreasing series of graphs from $T(G)$ to $T'(G)$. The largest positive difference in state space between any two graphs is $c(|u| + |v| - |u||v|)$ with $|u|$ or $|v|$ equal to 1. Therefore the largest increase in state space between graphs is c which will be less than or equal to $\mathcal{E}_{max}$. There are less than $|V|^2$ possible edges in a graph. □

In practice, $|V|^2$ is quite conservative as most of the nodes are typically not observed, and many of the desired non-minimal edges can be added through elimination.

Conjecture 20. *The above upper bound can be strengthened to use number of observations rather than $|V|^2$.*

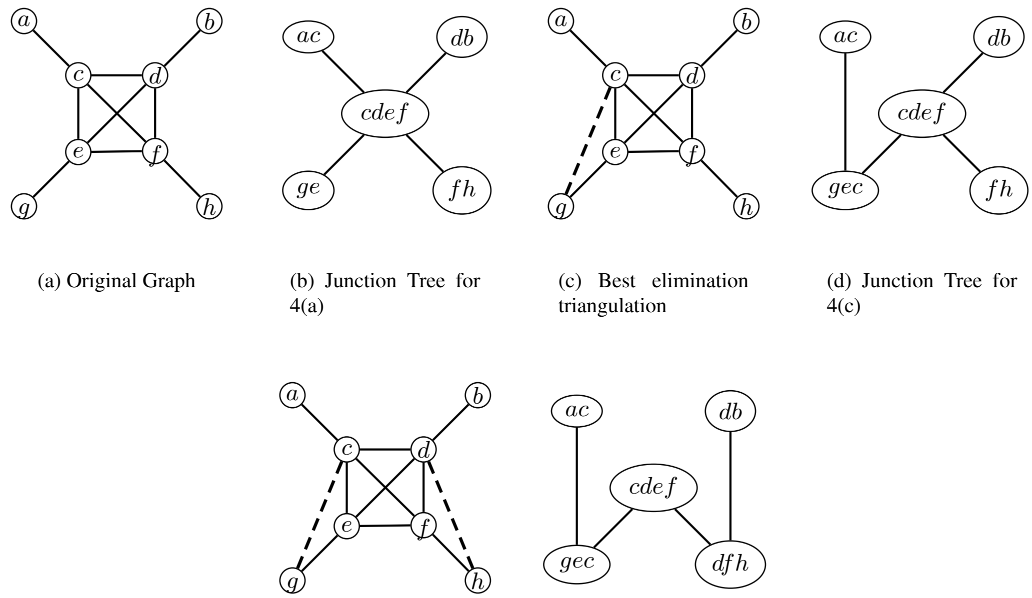
Although state space is the most common heuristic used to predict the computational requirements of inference, other heuristics have also been proposed which might better match a particular inference implementation. In [1] the following heuristic was proposed based on a detailed analysis of the specific number of additions and multiplications needed in their junction tree algorithm.

$$\sum_{c \in C} d(c)\mathrm{S}(c) - 2 \sum_{(c_1,c_2) \in Q} \mathrm{S}(c_1 \cap c_2)$$

Where C is the set of junction tree nodes (maximal cliques), $d(c)$ is the degree of the clique node c in the junction tree, Q is the set of edges in the junction tree, and $(c_1, c_2) \in Q$ is an edge in the junction tree between clique node c_1 and clique node c_2. Deterministic variables are not considered here. An example was given in [1] showing that increasing the number of variables in a clique can sometimes lower this value by allowing larger cliques to have a smaller degree in the junction tree. Increasing the number of variables in a clique is equivalent to adding non-minimal edges to the underlying triangulation. Here it is shown that elimination can not always create these triangulations which result in an optimal junction tree. The original graph is shown in Figure 4(a) with all variables having a cardinality of η. With no fill-in edges the resulting junction tree in Figure 4(b) has a score of $4\eta^4 + 4\eta^2 - 8\eta$. We can use elimination to obtain the triangulation in Figure 4(c) and lower the score to $3\eta^4 + 2\eta^3 - \eta^2 - 6\eta$. The optimal triangulation shown in Figure 4(e) has a score of $2\eta^4 + 4\eta^3 - 2\eta^2 - 4\eta$, and is not an elimination graph.

3.3 ELIMINATION AS MARGINALIZATION

Further insight into how elimination can affect the computation required for inference can be obtained by its comparison to the equivalent marginalizations of joint probability distributions [13]. Finding the marginal of a single variable



(a) Original Graph

(b) Junction Tree for 4(a)

(c) Best elimination triangulation

(d) Junction Tree for 4(c)

(e) Optimal triangulation

(f) Junction Tree for 4(e)

Figure 4: Non-elimination based triangulation gives improvement over elimination using heuristic from [1].

in a graphical model is done by integrating away all other variables in the distribution. We consider the distribution whose graph is shown in Figure 2. Summing the variables in order c, a, b, d gives the following equations:

$$
\begin{aligned}
p(e) &= \sum_d \sum_b \sum_a \sum_c p(a)p(b)p(c|a,d)p(d|a,b)p(e|b,d) \\
&= \sum_d \sum_b p(b)p(e|b,d) \sum_a p(a)p(d|a,b) \sum_c p(c|a,d) \\
&= \sum_d \sum_b p(b)p(e|b,d) \sum_a p(a)p(d|a,b)\phi_1(a,d) \\
&= \sum_d \sum_b p(b)p(e|b,d)\phi_2(b,d) \\
&= \sum_d \phi_3(e,d) = \phi_4(e)
\end{aligned}
$$

In general, marginalizing a vertex v in G creates a potential function $\phi(...)$ over variables $NE_G(v)$. Summing almost always precludes the possibility of further factorization of $\phi(...)$, and thus corresponds to completing the set $NE_G(v)$ in an elimination step. As another example, elimination ordering a, c, b, d yields the graph in Figure 2(b). One should also note that the complexity of performing the summations is on the order of the state space of the resulting triangulation.

One can also find the marginal of e through summation with a complexity on the order of the triangulation seen in Figure 2(d), but now summation over a vertex v does not necessarily create the analogous functions over $NE_G(v)$. One must either consider the summations as operations on a junction tree, on a transformed graph (Section 7.1.1), or the summation itself is intelligently aware of the determinism in the graph. In either case, we have:

$$\begin{aligned}p(e) &= \sum_{a,b,c,d} p(a)p(b)p(d|a,b)p(c|a,d)p(e|b,d) \\ &= \sum_{a,b,c} p(a)p(b)p(c|a,f(a,b))p(e|b,f(a,b)) \\ &= \sum_{b} p(b) \sum_{a} p(a)p(e|b,f(a,b)) \sum_{c} p(c|a,f(a,b))\end{aligned}$$

Here, we assume that $p(d|a,b) = \delta_{\{d=f(a,b)\}}$ where $f(a,b)$ is a deterministic function and $\delta()$is the Dirac delta function. As can be seen, this sequence of operations is not analogous to elimination on the original graph.

4 ELIMINATION GRAPH DETECTION

If we are given a graph and triangulation, one might be interested in determining if the triangulation is an elimination graph. In this section, a polynomial time algorithm is given for this purpose.

function **isEliminationGraph:**
  On input $\langle G = (V,E), T(G) = (V, E \cup F)\rangle$
  if $|V| = 1$ **then**
    return $true$
  else
    $A = \{v|(v$ simplicial in $T(G))$ & $(\text{NE}_G(v) = \text{NE}_{T(G)}(v))\}$
    if $A = \emptyset$ **then**
      return $false$
    else
      choose $v \in A$, return isEliminationGraph(G_v, $(T(G))_v$)
    end if
  end if

Lemma 21. *We are given a graph $G = (V,E)$ and a triangulation $T(G) = (V, E \cup F)$. Suppose F can be generated by some elimination order $\alpha = (\alpha(1), \alpha(2), .., \alpha(|V|))$. Also suppose that $\alpha(k)$ is simplicial in $T(G)$ and $\text{NE}_G(\alpha(k)) = \text{NE}_{T(G)}(\alpha(k))$. Then the order $\beta = (\alpha(k), \alpha(1), \alpha(2), ..., \alpha(k-1), \alpha(k+1), ..., \alpha(|V|))$ will also generate F.*

Proof. Consider eliminating $T(G)$ according to β. $\alpha(k)$ is simplicial in $T(G)$ so eliminating it will not add any edges to $T(G)$. Moreover, $\alpha(1)$ is simplicial in $T(G)$, so it will still be simplicial in $(T(G))_{\alpha(k)}$. Continuing with the nodes in β in turn, for each $i = 2, \ldots, k-1, k+1, \ldots, |V|$, $\alpha(i)$ will not have any neighbors in $T(G)$ that it does not also have at the point when, according to α, it is eliminated from $T(G)$. Therefore, β is a perfect ordering of $T(G)$. Define $\mathcal{E}_\beta(G) = (V, E \cup F_\beta)$. G is a spanning subgraph of $T(G)$ so if we eliminate both $T(G)$ and G according to β, from Lemma 9, $F_\beta \subseteq F$.

Now suppose that there is some edge $(v,w) \in F$ but $(v,w) \notin F_\beta$. $\text{NE}_G(\alpha(k)) \subseteq \text{NE}_{\mathcal{E}_\beta(G)}(\alpha(k)) \subseteq \text{NE}_{T(G)}(\alpha(k)) = \text{NE}_G(\alpha(k))$, so $v, w \neq \alpha(k)$. Define $S_\alpha = \{u \in V : \alpha(u) < \min[\alpha(v), \alpha(w)]\}$ and $S_\beta = \{u \in V : \beta(u) < \min[\beta(v), \beta(w)]\}$. If $k < \min[\alpha(v), \alpha(w)]$ then $S_\alpha = S_\beta$. If $k > \min[\alpha(v), \alpha(w)]$ then $S_\alpha = S_\beta \setminus \alpha(k)$. From Theorem 6, there is a path in G, $[v, v_1, v_2, ..., v_l, w]$, where $\alpha(v_i) \in S_\alpha \,\forall\, i = 1...l$ — but since $(v,w) \notin F_\beta$, no such path $[v, v_1, v_2, ..., v_m, w]$ exists in G such that $\beta(v_i) \in S_\beta \,\forall\, i = 1...m$. This, however, is a contradiction since $S_\alpha \subseteq S_\beta$. Therefore, we must have that $(v,w) \in F_\beta$. Therefore, $F_\beta = F$. ☐

Theorem 22. *isEliminationGraph will return $true$ if and only if F can be generated by some elimination order α.*

Proof. First it will be shown that if the algorithm returns true then $\exists \alpha$ such that $\mathcal{E}_\alpha = T(G)$. The proof is by induction on $|V|$. The theorem obviously holds for $|V| = 1$, assume it is true when $|V| = N-1$, and consider a graph where $|V| = N$. The first node chosen by the algorithm is v. The algorithm returned true on the original graph, so it also returns true on $\langle G_v,\ (T(G))_v\rangle$. From the induction hypothesis there exists $\mathcal{E}_{\beta(G_v)} = (T(G))_v$

so we construct elimination ordering α by concatenating v to the front of ordering β. $\mathrm{NE}_G(v) = \mathrm{NE}_{T(G)}(v)$ so $\mathrm{NE}_{\mathcal{E}_\alpha(G)}(v) = \mathrm{NE}_{T(G)}(v)$. The same holds for $V \setminus v$ from the induction hypothesis.

The proof of the forward direction is also by induction on $|V|$. The theorem obviously holds for $|V| = 1$, assume it is true when $|V| = N - 1$, and consider a graph where $|V| = N$. There is some elimination order α which generates F, but from Lemma 21 we can also rearrange the order such that any node, $\alpha(k)$ which is simplicial in $T(G)$ and $\mathrm{NE}_G(\alpha(k)) = \mathrm{NE}_{T(G)}(\alpha(k))$ is eliminated first. When the elimination graph $G_{\alpha(k)}$ is eliminated in the order $\beta \setminus \alpha(k)$ it will generate $T(G)_{\alpha(k)}$, and the theorem is proven by the induction hypothesis.

□

5 COMPLEXITY

It was proven in [2] that it is NP-complete to determine if a graph has a triangulation with treewidth k . It was proven in [25] that finding optimal state space triangulations is NP-hard through a reduction from the Elimination Degree Sequence problem. Here it is shown that the decision version of the problem is in NP for any polynomial time heuristic $f(T(G), I)$, where I is any information the heuristic needs to know about the vertices such as cardinality and *also* deterministic relationships. It is then shown that the state space problem is NP-complete by a reduction from the treewidth problem. This reduction is simpler than the reduction from the Elimination Degree Sequence problem.

Definition 23. *MAXTRI* $= \{\langle G = (V, E), I, \alpha\rangle \mid$ *G has a triangulation with* $f(T(G), I) < \alpha$ $\}$

Theorem 24. *The MAXTRI problem is in NP for all polynomial* $f(G, I)$.

Proof. The following program can verify an member of MAXTRI with a choice of fill in F:

V ="On input $\langle V, E, I, F, \alpha\rangle$:

1. Build the graph $G = (V, E \cup F)$
2. Check if G is triangulated
3. If $f(G, I) < \alpha$ *accept*; otherwise *reject*"

The size of F is $< |V|^2$, testing if $G = (V, E \cup F)$ is triangulated can be done in polynomial time [23], and testing that $f(G, I) < \alpha$ is polynomial time by definition. □

Definition 25. *MAXTREEWIDTH* $= \{\langle G = (V, E), k\rangle \mid$ *G has a triangulation with treewidth* $\leq k$ $\}$

Theorem 26. *MAXTREEWIDTH is NP-complete. [2]*

Definition 27. *MAXSTATESPACE* $= \{\langle G = (V, E), I, \alpha\rangle \mid$ *G has a triangulation with state space* $< \alpha\}$

Theorem 28. *MAXSTATESPACE is NP-complete.*

Proof. First it is shown that MAXTREEWIDTH is polynomial time reducible to MAXSTATESPACE. Create an I assigning each vertex a cardinality of $|V|$ and defining no deterministic dependencies, and return the result of MAXSTATESPACE($G, I, |V|^{k+2}$). There are at most $|V| - 1$ maximal cliques in a triangulated graph, so if $T(G)$ has a treewidth $\leq k$ this construction will give a state space less than $|V||V|^{k+1} = |V|^{k+2}$, and MAXSTATESPACE will accept. If the smallest treewidth for G is $\geq k + 1$ the smallest possible state space is $|V|^{k+2}$ and MAXSTATESPACE will reject. Similarly, if MAXSTATESPACE accepts the treewidth is at most k and if it rejects the treewidth is $> k$. From [11, section 2] the maximal cliques of a triangulated graph can be found in polynomial time, and given the maximal cliques the state space can be calculated in polynomial time. From Theorem 24, MAXSTATESPACE is in NP, therefore MAXSTATESPACE is NP-complete. □

6 METHODS

In this section, we describe methods designed for finding non-minimal triangulations with the goal of finding optimal triangulations in graphs containing a mix of random and deterministic nodes. The first method, pre-elimination, is the basis for a number of heuristics for finding good triangulations in such graphs. The second method is a stochastic search over the space of all possible triangulations. Finally, we describe two additional novel approaches that, while not yet implemented and tested, are presented herein for comparison and completeness purposes.

6.1 ELIMINATION WITH EXTRA EDGES

Unlike elimination, any method with the capability of finding all possible triangulations will certainly be able to find the optimal one. In this section, we thus start with an algorithm with such a capacity. The algorithm is presented in its most general form here, and the following subsections use it to find small state space triangulations.

Definition 29. Elimination with pre-elimination edge addition (pre-elimination): *Elimination proceeds as usual, but in addition any number of extra fill-in edges can be added between uneliminated vertices.*

We define pre-elimination formally by giving a new definition for deficiency. We have a graph $G = (V, E)$, elimination ordering $\alpha : \{1, 2, ..., |V|\} \leftrightarrow V$, and for each $v \in V$ a set of extra edges $H_v \subseteq \{(u_1, u_2) \mid (u_1, u_2) \notin E, u_1 \neq u_2, \alpha^{-1}(v) \leq \alpha^{-1}(u_1), \alpha^{-1}(v) \leq \alpha^{-1}(u_2)\}$, $H = \bigcup_{i=1}^{|V|} H_{\alpha(i)}$. When $H = \emptyset$, Definition 30 reduces to the original Definition 1, and pre-elimination reduces to elimination.

Definition 30. *The* **pre-deficiency** *of a vertex v in G with extra edges H is: $D_{G,H}(v) = (\{u, w\} \mid \{v, u\} \in E \cup H, \{v, w\} \in E \cup H, \{u, w\} \notin E \cup H\})$.*

Pre-elimination allows extra edges to be added at any point during the elimination process. For example, one could add a particular edge to the original graph before any eliminations occur, or one can eliminate several vertices and then add the edge to the resulting v-elimination graph. The following theorem states that the same triangulation will result, regardless of when extra edges are added.

Lemma 31. *In pre-elimination the extra fill-in edges can be added at any point, from before any node is eliminated to just before the first of the edge's endpoints is eliminated, and the same fill-in will result.*

Proof. The deficiency depends directly on the union of all extra edges, H, and not on the placement of the edges in a particular H_v for different v since any change therein will not affect the union H. Consider next the result of extra edges when eliminating a node v. Any edge in H_{v_k} where $\alpha(v_k) < \alpha(v)$ will have been removed from the graph when v is eliminated. Suppose we have an edge $f \in H_{v_l}$ where $\alpha(v_l) > \alpha(v)$. If f connects two neighbors of v, then f is added anyway when v is eliminated. If f connects a neighbor of v to a non-neighbor of v, it will not change the neighbors of v and the fill-in resulting from the elimination of v is unaltered by the existence of f. If f connects two non-neighbors of v, again, the pre-deficiency of v is unchanged. □

Theorem 32. *Pre-elimination always results in a triangulated graph.*

Proof. From Theorem 31 we can add all the extra edges before any nodes are eliminated and follow this with normal elimination, and elimination always results in a triangulated graph. □

Theorem 33. *In pre-elimination, if an extra edges is added to a node just before the node is eliminated then the extra edge will be non-minimal. Or, all $(u, v) \in H_v$ are non-minimal.*

Proof. Nodes can not be in an unchorded cycle involving a node eliminated before it. Suppose v is eliminated and extra edge (u, v) chords some four cycle v, n_1, u, n_2, v. n_1 and n_2 are neighbors of v and are connected, so the cycle is chorded without (u, v). □

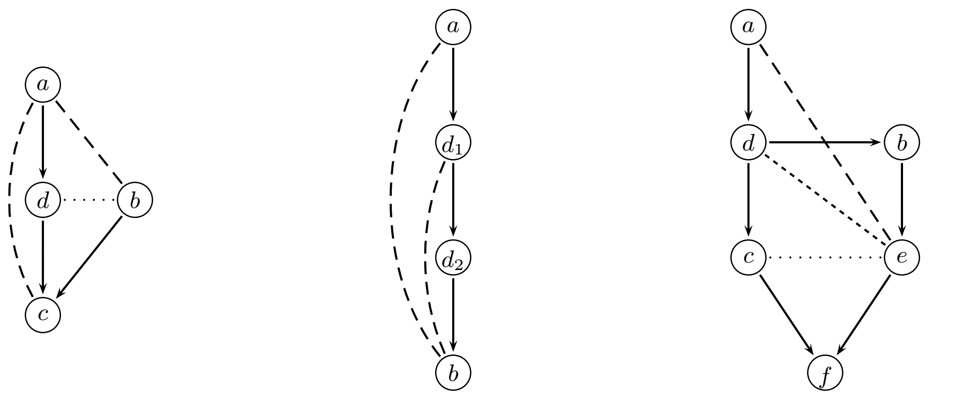
Note that if an extra edge is added earlier in the triangulation, it might have been added anyway due to some elimination step, and therefore could be minimal.

Theorem 34. *Pre-elimination can create any triangulation.*

Proof. To obtain $T(G) = (V, E \cup F)$, add all edges in F and eliminate according to some perfect elimination order. □

6.1.1 Ancestral Edges

In the general case elimination with extra edges has a search space much worse than examining all choices of fill-in, but when optimizing for state space all possible extra edges do not need to be considered. Instead, we only have to choose from the set of extra edges that ensure that deterministic variables are placed in cliques with their parents. The graphs considered in this work are moralized Bayesian networks where the value of a deterministic variable is



(a) (a, c) is ancestral from child c, and (a, b) is ancestral because b is a neighbor from moralization

(b) (d_1, b) is ancestral with respect to d_2, (a, b) is ancestral with respect to d_1 because of ancestral edge (d_1, b)

(c) (d, e) is fill-in, which causes (a, e) to be ancestral

Figure 5: Ancestral edges with respect to the deterministic nodes, labeled d

defined by a function of its parents. In such a graph, moralization will cause each deterministic variable to be in at least one maximal clique with its parents, but the variable might also be a member of other maximal cliques due to its non-parent neighbors. These neighbors are the deterministic node's children, any undirected neighbors gained during the moralization step, or any undirected neighbors needed to triangulate the graph. In certain contexts there might be neighbors from other sources as well, such as when triangulating a DBN additional fill-in edges are added to make the boundaries between frames a complete set [5]. We next define a type of edge that will be particularly useful.

Definition 35. *For any deterministic node n, an* **ancestral edge** *with respect to n is one which connects a parent of n to a child or undirected neighbor of n. An ancestral edges is any edge that is ancestral with respect to some deterministic n. See Figure 5.*

It will now be shown that the optimal state space triangulation can always be found using pre-elimination where we limit the choice of extra edges to the set of fill-in edges that will be ancestral. We note here, however, that it is not sufficient for optimal state space to simply add all ancestral edges to the graph (See Figures 6a,b).

Lemma 36. *Consider a graph G with node cardinalities ≥ 2 and triangulation $T(G)$. Suppose an edge (p, c) is added to form triangulation $T_{new}(G)$. If $S(T_{new}(G)) < S(T(G))$ then (p, c) is ancestral with respect to some deterministic node n.*

Proof. Lemma 15 gives that there is only one maximal clique containing (p, c). From lemma 16, the state space can not decrease if the maximal clique in $T_{new}(G)$ which contains p and c is made up only of random variables. The maximal clique in $T_{new}(G)$ containing p and c must also have some deterministic variable, n which contributes to the state space in $T(G)$ but does not contribute in $T_{new}(G)$. That is, there are maximal cliques $C_1, C_2 \in T(G)$ where $p \in C_1$, $c \notin C_1$, $c \in C_2$, $p \notin C_2$, and $n \in C_2$, so that the addition of (p, c) creates a maximal clique $C_{new} = C_1 \cup C_2$ and $p, n, c \in C_{new}$. Node c can not be a parent of n or else (p, c) would have existed in $T(G)$ from moralization. Therefore, (p, c) is ancestral with respect to deterministic node, n. □

Theorem 37. *Elimination with pre-elimination edge addition where the extra edges are limited to ancestral edges is sufficient to find an optimal state space triangulation when all cardinalities are ≥ 2.*

Proof. We are given a graph $G = (V, E)$ and desire to obtain an optimal state space triangulation $T(G) = (V, E \cup F)$. First, define the set of edges, A, as any edge $\in F$ which is ancestral in $T(G)$. Next, construct $G' = (V, E \cup A)$. $T(G)$ is a triangulation of G' with fill-in $F' = F \setminus A$. Next we want to conclude that F' forms a minimal triangulation of G'. Assume that this conclusion is not true, and there is a set of edges $M \subset F'$ such that $T_{min}(G') = (V, (E \cup A) \cup M)$ is triangulated. From Lemma 7 there is an increasing sequence of graphs which differ by one edge beginning with

$T_{min}(G')$ and ending with $T(G)$. $S(T(G))$ is optimal so $S(T_{min}(G')) \geq S(T(G))$. If $S(T_{min}(G')) > S(T(G))$ at least one graph in the sequence of graphs from $T_{min}(G')$ to $T(G)$ must have a lower state space than the previous graph. From Lemma 36, this is a contradiction because F' does not contain any ancestral edges. Therefore, F' is a minimal triangulation of G' and $T(G)$ is an elimination graph of G'. If $S(T_{min}(G')) = S(T(G))$ we have not shown a contradiction but $T_{min}(G')$ is also an optimal triangulation and is an elimination graph of G'. In either case, all extra edges are ancestral. □

We leave it as an open problem to determine if there are other conditions which define a smaller set of edges that are necessary for finding optimal state space triangulations.

6.1.2 Ancestral Edge Heuristics

Pre-elimination with ancestral edges gives a framework for finding triangulations in networks with deterministic dependencies, but gives no guidance on which ancestral edges should be chosen. This section describes a pre-processing step which adds ancestral edges to a graph, and this new graph is then triangulated using standard elimination heuristics. The heuristics given here will only choose edges from what we will define as the original graph ancestral edges. An original graph ancestral edge is a fill-in which would be ancestral when when added to the original graph, or ancestral after other original graph ancestral edges have been added.

Definition 38. (u_k, v_k) *is an* **original graph ancestral edge** *of* $G = (V, E)$ *if* $(u_k, v_k) \notin E$ *and there is a sequence of graphs* $G_1 = (V, E \cup \{(u_1, v_1)\})$, $G_2 = (V, E \cup \{(u_1, v_1), (u_2, v_2)\})$, *...,*$G_k = (V, E \cup \{(u_1, v_1), (u_2, v_2), .., (u_k, v_k)\})$ *where* (u_i, v_i) *is ancestral in* $G_i, i = 1...k$.

It is an open problem to determine a bound on the number original graph ancestral edges. It is at least $O(2^{|V|})$, but there are no examples showing that the number of original graph ancestral edges can grow as $O(2^{|V|^2})$. Considering only original graph ancestral edges does not consider all possible ancestral edges. The fill-in due to an elimination step could increase the potential number of ancestral edges. To consider all possible ancestral edges one would need to determine the original graph ancestral edges with respect to each possible elimination graph. Such a technique would require more computation, but might give better triangulations.

Three heuristics for deciding which extra edges to add are proposed here. The first is to use all original graph ancestral edges, called pre-all. The next is to randomly select a subset of original graph ancestral edges, called pre-random. The last heuristic is to choose the original graph extra edges which are locally optimal, called pre-lo. That is, if we have a deterministic node, v, with parents pa(v) and child or neighbor c, the set of edges between c and pa(v) is locally optional if $S(c \cup v \cup \text{pa}(v)) < S(c \cup v) + S(v \cup \text{pa}(v))$.

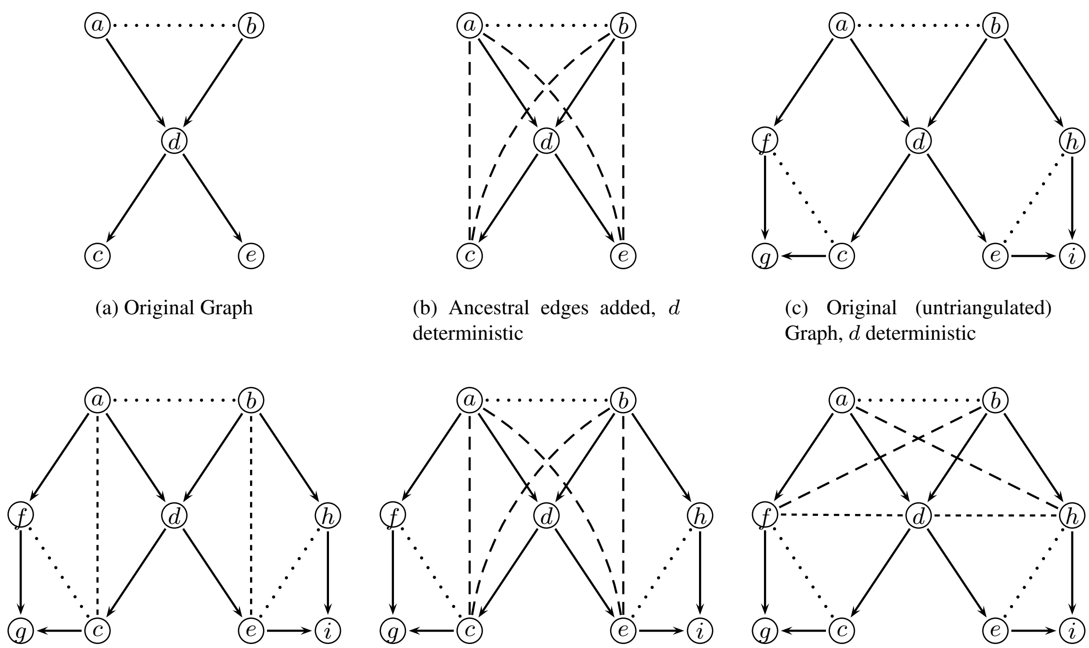
6.1.3 Weaknesses of the Heuristics

It might at first seem that either the pre-all or the pre-lo heuristic can always find the optimal augmented graph, transforming the problem into a standard elimination search. This section seeks to demonstrate why these heuristics (or some equivalent graph transformation) will not always work. The main difficulty is that the decision to add an ancestral edge can not be made optimally without considering the entire triangulation.

In Figure 6(a) with d deterministic the choice of adding ancestral edges will depend only on the cardinality of d. Suppose $|a| = |b| = |c| = |e| = 10$ and $|d| = 40$, then the graph will have a state space of 900 without any fill-in and a state space of 2000 when, as is Figure 6(b), all ancestral edges have been added. In this case it is easy to see that no ancestral edges should be added and any perfect elimination order can be used.

In Figure 6(c) with d again deterministic, the decision is not so simple. Either (a, c) or (f, g) and either (b, e) or (d, h) needs to be added in order to triangulate the graph. Just as in Figure 6(a) it is not locally optimal to add the ancestral edges, but if (a, c) and (b, e) are added to triangulate the graph, as in Figure 6(d), the ancestral edges might now be beneficial. If the cardinality of all of the random variables is 10 and the cardinality of d is 40, then Figure 6(d) has a state space of 12100, but Figure 6(e) which includes the ancestral edges has a state space of only 6000.

Now suppose $|f| = |h| = 2$ and as before $|d| = 40$ and the others have a cardinality of 10. Now the best state space triangulation is seen in Figure 6(f). The edges (b, f) and (a, h) are not original graph ancestral edges, but become ancestral only after (f, d) and (d, h) are added. This triangulation can not be found through the given heuristics, and could only be found by an algorithm which examines the possible ancestral edges after every elimination step.



(a) Original Graph

(b) Ancestral edges added, d deterministic

(c) Original (untriangulated) Graph, d deterministic

(d) Minimal fill-in

(e) Ancestral edges added

(f) Non-original graph ancestral edges

Figure 6: Graphs illustrating the weaknesses of the extra edge heuristics

Table 1: Results on real-world graphs. The first three columns are the state spaces of the prologue, chunk, and epilogue partitions. The fourth column is the state space when these triangulations are used to unroll the graph to 100 frames. State spaces are given in $\log_{10}$. The fourth column is inference time in seconds.

	P	**C**	**E**	**100x**	**Seconds**
Multi-Stream Training					
elim. only	2.94	4.46	4.60	6.46	894.6
pre-edge all	**0.90**	**4.05**	4.65	6.07	549.9
pre-edge random	1.00	4.06	4.67	6.08	586.3
pre-edge local	1.41	4.24	4.56	6.25	585.4
completed	**0.90**	**4.05**	4.65	6.07	557.2
edge anneal	0.96	4.06	**4.19**	**6.06**	**403.8**
elim. anneal	2.54	4.23	4.32	6.24	477.9
Aurora Training					
elim. only	2.06	2.96	3.29	4.97	56.8
pre-edge all	**0.90**	**2.85**	**3.15**	**4.86**	50.3
pre-edge random	1.00	**2.85**	3.25	4.86	50.0
pre-edge local	1.15	3.02	3.25	5.03	**47.7**
completed	**0.90**	**2.85**	**3.15**	**4.86**	50.1
edge anneal	0.95	2.89	3.07	4.90	50.1
elim. anneal	2.07	2.97	3.16	4.98	55.9
Edit Distance Training					
elim. only	1.41	5.09	5.39	7.10	156.7
pre-edge all	1.41	5.09	6.49	7.19	157.3
pre-edge random	1.41	5.09	5.40	7.10	157.7
pre-edge local	1.41	5.09	**5.39**	7.10	157.1
completed	**1.40**	**5.09**	6.49	7.19	157.3
edge anneal	**1.40**	**5.09**	5.39	**7.10**	**157.1**
elim. anneal	**1.41**	5.09	5.39	7.10	157.8

6.2 GENERAL PURPOSE STOCHASTIC SEARCH

In the case when a graph is used many times, used on large data sets, or when fast heuristic searches do not provide an adequate solution, a stochastic search is often desirable. In a stochastic search one wishes to move around the search space in a manner such that each move results in is a small change from the previous state, and all states are reachable through such a series of moves.

Theorem 39. *Given any two triangulations of a graph, $T_a(G)$ and $T_b(G)$ there is a sequence of triangulated graphs $T_a(G), T_1(G), T_2(G), ..., T_b(G)$ with only a single edge difference between subsequent graphs in the sequence.*

Proof. Suppose both $T_a(G)$ and $T_b(G)$ are spanning subgraphs of $T_s(G)$, from Lemma 7 one can create a series of graphs from $T_a(G)$ to $T_s(G)$ and from Lemma 8 create a sequence from $T_s(G)$ to $T_b(G)$. All triangulations of G are spanning subgraphs of the triangulation formed by completing the set of all vertices. □

In Section 7, this method is used inside a simulated annealing algorithm, similar to the elimination annealing algorithm given in [14]. An ancestral edge that is missing in the original graph is chosen randomly. If the edge exists in the current triangulation it is removed, and if it does not exist it is added. If the resulting graph is not triangulated the change is undone and a new edge is chosen. The state space of the resulting triangulation is calculated and it is accepted or rejected based on the state space of the previous triangulation and the current annealing temperature.

7 RESULTS

We present results for two types of graph. First, we applied our results on a set of dynamic Bayesian networks (DBNs), consisting of mixed random/deterministic variables, many of which are used for real-world speech recognition and language tasks. Such dynamic networks often crucially rely on deterministic implementations to make the models

Table 2: Results on real-world graphs. The first three columns are the state spaces of the prologue, chunk, and epilogue partitions. The fourth column is the state space when these triangulations are used to unroll the graph to 100 frames. State spaces are given in $\log_{10}$.

	P	**C**	**E**	**100x**
Phone Free Decoding				
elim. only	6.92	9.84	8.80	11.8
pre-edge all	**2.21**	10.41	12.32	12.7
pre-edge random	**2.21**	9.78	9.58	11.8
pre-edge local	**2.21**	9.84	8.61	11.8
completed	5.82	13.21	18.55	18.6
edge anneal	2.36	8.77	8.79	10.8
elim. anneal	2.39	**8.77**	**8.30**	**10.8**
Ears Decoding				
elim. only	6.86	9.04	8.99	11.0
pre-edge all	**6.70**	9.58	9.88	11.6
pre-edge random	**6.70**	9.58	**9.84**	11.6
pre-edge local	**6.70**	**9.03**	8.99	11.0
completed	**6.70**	9.58	9.88	11.6
edge anneal	**6.70**	9.03	8.90	**11.0**
elim. anneal	6.71	9.03	8.90	11.0

they express tractable. The second set of results come from a set of randomly generated static Bayesian networks with mixed deterministic and random node implementations.

The first set of results, given in Tables 1 and 2, examine the performance of the heuristics on real-world speech recognition and string DBNs. Multi-Stream Training [26, based on Figure 2] and Aurora Training [10] are used in digit recognition systems. Edit Distance Training is used to train costs for Levenstein distance and contains 8 variables per frame. Phone Free Decoding is used in an isolated word recognition system. This graph has 40 variables per frame, 11 of which are observed. Ears Decoding [1] is a graph for word recognition with cross-word dependencies and has 11 variables per frame. The DBNs are divided into three partitions: a prologue (P), chunk (C), and epilogue (E) as described in [5]. Results are given for each graph partition, the graph unrolled 100 times, and in Table 1, inference wall-clock time on real (but a small set of) data. All results are the base 10 logarithm of the state space. The label 'elim. only' is the best triangulation found after multiple iterations of a number of standard elimination heuristics. These include minimum fill, minimum size, minimum weight [14], maximum cardinality search [24], random orderings, and various combinations of these. Pre-all, pre-random, and pre-lo are the pre-processing steps described in Section 6.1.2 followed by the same suite of elimination heuristics. Completed is combining the entire chunk into a single clique, 'edge anneal' is the technique described in Section 6.2, and 'elim. anneal' is annealing over elimination orders as described in [14].

In Multi-Stream Training the extra edge heuristics see a significant performance increase over the standard heuristics, and edge annealing outperformed elimination annealing. In Aurora Training the techniques not limited by elimination outperformed the elimination techniques, but to a lesser degree. Improvements were not seen in Phone Free Decoding and Ears Decoding. The last two graphs (Table 2) were not timed because the memory requirements of standard methods for exact inference are too large, but inference could be performed with these triangulations when pruning and/or sampling techniques are applied.

Next, 50 static (non-DBN) graphs were randomly generated for the next set of results. The graphs have 30 variables, a maximum of 87 edges, and variables have a 0.5 probability of being deterministic. Hidden variables have cardinalities of 2 to 5, observed have cardinalities of 50, and deterministic have cardinalities from 2 to the product of their parents' cardinality with a maximum of 125. No preprocessing, pre-all, pre-random, and pre-lo are followed with elimination using minimum fill, minimum weight [14], random, and maximum cardinality search. Elimination and edge annealing were also performed. Each heuristic was given five minutes of running time; the annealing methods needed many hours to complete. Inference was performed using random data and the results were quite dramatic. Often the triangulations were intractable without the ancestral edge step, but ran in a few seconds with it. For each

[1] thanks to Karim Filali, Karen Livescu, and Ozgur Cetin

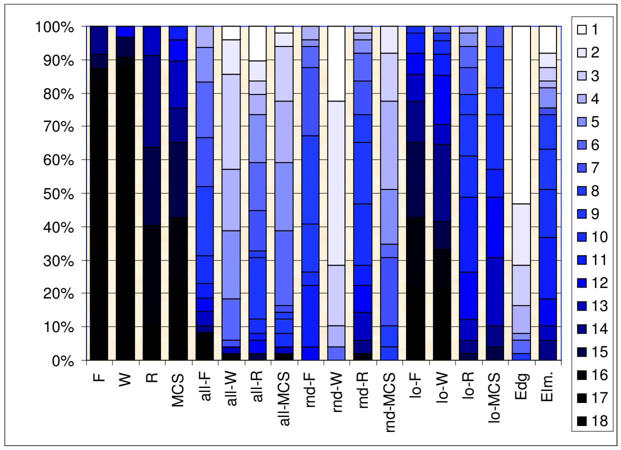


Figure 7: Percentage of graphs where heuristic scored a particular ranking. Key: F=minimum fill, W=minimum weight, R=random, MCS=maximum cardinality search, all=pre-all, rnd=pre-random, lo=pre-lo, Edg.=edge annealing, and Elim.=elimination annealing.

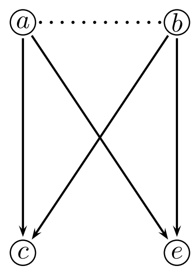


Figure 8: whee Transformation of Figure 2

graph the state space of each method was ranked, and the chart in Figure 7 shows the percentage of time each method gave the best score, second best score, and so on. Edge annealing gave the best result on 52% of the graphs. The best heuristic methods were pre-random and pre-all followed by minimum weight.

7.1 OTHER METHODS

This section describes two alternative approaches that in order to compare with those that were implemented and tested above. These approaches do not appear to have any fundamental advantages over the approaches used in this work, but might yield effective results if implemented and tested experimentally and are moreover interesting in their own right to contrast against the methods above.

7.1.1 Graph Transformations

One alternative to adding ancestral edges is to instead transform the graph into one which does not include the deterministic variables, and then make use of a minimal triangulation. Such a strategy would work on the graph in Figure 2 and is illustrated in Figure 8. However, this does not make the search problem any easier then our fill-in approach because in the general case one would not know the complete benefit of a graph transformation without considering the entire triangulation. The problem is the same as choosing ancestral edges, and as seen in Section 6.1.3 there does not appear to be a simple solution for choosing them. One could devise a method which combines a search over the possible transformations and also the minimal triangulations of these graphs, but this does not appear to have any computational or conceptual advantages over the more uniform approach of searching all choices of fill-in. A graph transformations also has the disadvantage in that it might remove variables which are useful to the graph designer. For example, one might want to know the most likely assignments of a set of variables, some of which include nodes that have been transformed out of the graph — staying in the realm of triangulation by edge fill in would require no change to an inference algorithm in order to extract such information.

7.1.2 Post-Elimination Ancestral Edges

One could also find optimal triangulations by adding extra edges to a graph after a triangulation is found through elimination. The advantage of this method is that one would know the location of all of the fill-in edges and hence the location of all the ancestral edges. The biggest disadvantage is that the addition of some ancestral edges might cause the graph to no longer be triangulated and this would need to be monitored. One could also effectively add post-elimination edges though some algorithm which combines the nodes of a junction tree. The main difficulties are determining how to choose the initial triangulation and taking into consideration that there are many possible junction trees for a given triangulation. Again, post-elimination edge addition would be an interesting topic for future work and might prove effective, but does not offer any immediately obvious advantages in reducing the search space.

8 CONCLUSION

This paper gave an algorithm for the detection of triangulations which are not elimination graphs. It also gave a new prove that deciding if a graph has a triangulation with state space beneath a threshold is NP-complete. It was shown that elimination alone is not adequate for finding triangulations in many graphs which have deterministic dependencies, and heuristic methods and a stochastic search method for finding better solutions were given. Future work will be done to find a tighter bound for how close to optimal elimination in the presence of observations can come to the minimal state space solution. Work will also be done to determine if constraints can be found to limit extra-edges to sets smaller than the the ancestral edges, analyze the effects of examining the potential ancestral edges at every elimination step, and to evaluate the heuristics suggested in Sections 6.1.3 and 7.1.

Acknowledgments

This work was supported by NSF grant IIS-0093430 and an Intel Corporation Grant.

References

[1] S. M. Aji and J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46:325–343, 2000.

[2] S. Arnborg, D. J. Corneil, and A. P. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Alg. Dis. Meth.*, 8:227–284, 1987.

[3] C. Bartels and J. Bilmes. Elimination is not enough: Non-minimal triangulations for graphical models. submitted to *Eighteenth Annual Conference on Neural Information Processing Systems*, 2004.

[4] A. Becker and D. Geiger. A sufficiently fast algorithm for finding close to optimal junction trees. In *Proceed. 12th Conf. on Uncertainty in AI*, 1996.

[5] J. Bilmes and C. Bartels. On triangulating dynamic graphical models. In *Proceed. 19th Conf. on Uncertainty in Artificial Intelligence*, Acapulco, Mexico, 2003.

[6] H. L. Bodlaender, Arie M.C.A Koster, F. van den Eijkhof, and L. C. van der Gaag. Pre-processing for triangulation of probabilistic networks. In *Proceedings of the 17th Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 32–39, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[7] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in bayesian networks. In *Proceed. 12th Conf. on Uncertainty in Artificial Intelligence*, Portland, Oregon, 1996.

[8] R. Dechter. Bucket elimination: a unifying framework for probabilistic inference. In Michael I. Jordan, editor, *Learning and inference in graphical models*, pages 75–104. MIT Press, Cambridge, MA, 1998.

[9] R. Dechter and D. Larkin. Proceedings of the 17th annual conference on uncertainty in artificial intelligence. In *Hybrid processing of belief and constraints*, pages 112–119. Morgan Kaufmann Publishers, San Francisco, CA, 2001.

[10] J. Bilmes et. all. Discriminatively structured graphical models for speech recognition: JHU-WS-2001 final workshop report. Technical report, CLSP, Johns Hopkins University, Baltimore MD, 2001.

[11] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM J. Comp.*, 1(2):180–187, June 1972.

[12] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, 1980.

[13] M.I. Jordan and C. M. Bishop, editors. *An Introduction to Graphical Models*. to be published, 200x.

[14] U. Kjaerulff. Triangulation of graphs-algorithms giving small total state space. Technical Report R-90-09, Aalborg University, March 1990.

[15] D. Larkin and R. Dechter. Bayesian inference in the presence of determinism. In *Proc. of AISTATS,'03*. Soc. for AI and Statistics, Key West, Fl, 2003.

[16] P. Larraaga, C. Kuijpers, M. Poza, and R. Murga. Decomposing bayesian networks by genetic algorithms. *Statistics and Computing*, 7(1):19–34, 1997.

[17] S. Lauritzen. *Graphical Models*. Oxford Science Publications, 1996.

[18] S. Lauritzen and D. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *The Journal of the Royal Statistical Society*, 50:57–224, 1988.

[19] M. Meila and M. I. Jordan. Triangulation by continuous embedding. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9. The MIT Press, 1997.

[20] T. Ohtsuki, L. K. Cheung, and T. Fujisawa. Minimal triangulation of a graph and optimal pivoting order in a sparse matrix. *J. Math. Anal. Appl.*, 54:622–633, 1976.

[21] A. Parra and P. Scheffler. How to use the minimal separators of a graph for its chordal triangulation. In *Automata, Languages and Programming*, pages 123–134, 1995.

[22] D. J. Rose. Triangulated graphs and the elimination process. *J. Math. Anal. Appl.*, 32:597–609, 1970.

[23] D. J. Rose, R. E. Tarjan, and G. S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM J. Comp.*, 5:266–283, 1976.

[24] R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM J. Comp.*, 13:566–579, 1984.

[25] Wen Wilson. Optimal decomposition of belief networks. In *Proceedings of the 6th Annual Conference on Uncertainty in Artificial Intelligence*, pages 209–224, New York, NY, 1991. Elsevier Science Publishing Company, Inc.

[26] Y. Zhang, Q. Diao, S. Huang, W. Hu, C. Bartels, and J. Bilmes. DBN based multi-stream models for speech. In *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Hong Kong, China, April 2003.