# On the Importance of Elimination Heuristics in Lazy Propagation

Anders L. Madsen
HUGIN EXPERT A/S, Denmark
Department of Computer Science, Aalborg University, Denmark
alm@hugin.com

Cory J. Butz
Department of Computer Science, University of Regina, Canada
butz@cs.uregina.ca

**Abstract**

Belief update in a Bayesian network using *Lazy Propagation* (LP) proceeds by message passing over a junction tree (JT). In the process of computing a message, a set of variables is eliminated. As the JT provides only a partial order on the elimination of variables, it is necessary to identify elimination orders on-line. This paper considers the importance of elimination heuristics in LP when using *Variable Elimination* (VE) as the message and single marginal computation algorithm. It considers well-known cost measures for selecting the next variable to eliminate and a new cost measure. The empirical evaluation examines different heuristics as well as sequences of cost measures, and was conducted on real-world and randomly generated Bayesian networks. The results show that for most cases performance is robust relative to the cost measure used and in some cases the elimination heuristic can have a significant impact on performance, especially for JTs that are non-optimal.

## 1 Introduction

A *Bayesian network* (BN) (Pearl, 1988) is a powerful and popular model for probabilistic inference. Its graphical nature makes it well-suited for representing complex problems where the interactions between entities represented as variables can be described using *conditional probability distributions* (CPDs). Since the computational complexity of both exact and approximate probabilistic inference in BNs are NP-hard (Cooper, 1990b) and (Dagum and Luby, 1993), we have to rely on methods that in the worst case have exponential complexity (unless P=NP). To improve feasibility of probabilistic inference using such methods, it is important to consider extensions of existing methods that can lead to better performance.

There exist two main classes of algorithms for probabilistic inference in a BN: algorithms based on message passing in a secondary computational structure, i.e., a JT, (Lauritzen and Spiegelhalter, 1988; Jensen et al., 1990; Shenoy and Shafer, 1990) and direct algorithms operating on the CPDs associated with the BN, including belief propagation (Pearl, 1982; Kim and Pearl, 1983), VE (Zhang and Poole, 1994), the peeling method (Cannings et al., 1978), arc-reversal (Olmsted, 1983; Shachter, 1986), Recursive Decomposition (Cooper, 1990a) and Symbolic Probabilistic Inference (SPI) (Shachter et al., 1990; Li and D'Ambrosio, 1994; Bloemeke and Valtorta, 1998). LP (Madsen and Jensen, 1999) is a hybrid algorithm combining Shenoy-Shafer propagation and direct algorithms with the aim of exploiting independence properties induced by evidence and barren variables (Shachter, 1986).

This paper considers the importance of elimination heuristics in LP when using VE as the message and single marginal computation algorithm. The importance is assessed by an empirical evaluation, including both real-world and randomly generated BNs. A total of six different heuristics for selecting the next variable to eliminate are considered along with sequences of cost measures.

The experimental results confirm the assumption that the triangulating order can have a large impact on performance and that the effectiveness of using multiple cost measures depends on the structure of the network and its JT. Multiple cost measures appear to be most effective on non-optimal JTs, and less so on (near) optimal JTs.

## 2 Preliminaries and Notation

Here preliminaries and notation are introduced.

### 2.1 Bayesian Networks

Let $\mathcal{X} = \{X_1, \ldots, X_n\}$ be a set of discrete random variables such that $\text{dom}(X)$ is the state space of $X$ and $||X|| = |\text{dom}(X)|$. A discrete BN $\mathcal{N} = (\mathcal{X}, G, \mathcal{P})$ over $\mathcal{X}$ consists of an acyclic directed graph (DAG) $G = (V, E)$ with vertices $V$ and edges $E$ and a set of CPDs $\mathcal{P} = \{P(X \,|\, \text{pa}(X)) : X \in \mathcal{X}\}$, where $\text{pa}(X)$ denotes the parents of $X$ in $G$ (Pearl, 1988; Cowell et al., 1999; Kjærulff and Madsen, 2008). The BN $\mathcal{N}$ is an encoding of a joint probability distribution over $\mathcal{X}$

$$P(\mathcal{X}) = \prod_{i=1}^{n} P(X_i \,|\, \text{pa}(X_i)).$$

Belief update in $\mathcal{N}$ is defined as the task of computing the posterior marginal $P(X \,|\, \epsilon)$, for each non-evidence variable $X \in \mathcal{X} \setminus \mathcal{X}_\epsilon$ given a set of variable instantiations $\epsilon$, where $\mathcal{X}_\epsilon \subseteq \mathcal{X}$ is the set of variables instantiated by $\epsilon$.

A potential on $\text{dom}(\phi) = \mathcal{Y}$ is a function $\phi$ such that $\phi(y) \geq 0$, for each configuration $y \in \text{dom}(\mathcal{Y})$ and at least one $\phi(y)$ is positive (Shafer, 1996). The *domain graph* $G(\{\phi\}) = (V, E)$ induced by a potential $\phi$ is defined as the graph over $V = \text{dom}(\phi)$ with edges $E = \{(H_1, H_2), (H_2, H_1) \mid H_1, H_2 \in \text{head}(\phi)\} \cup \{(T, H) \mid H \in \text{head}(\phi), T \in \text{tail}(\phi)\}$ where $\text{head}(\phi)$ and $\text{tail}(\phi)$ are the conditioned and conditioning variables of $\text{dom}(\phi)$, respectively. That is, $G$ contains an undirected edge $(H_1, H_2)$ for each pair $H_1, H_2 \in \text{head}(\phi)$ and a directed edge $(T, H)$ for each pair $T \in \text{tail}(\phi)$ and $H \in \text{head}(\phi)$ (Madsen, 2006).

The domain graph of a set of potentials $\Phi = \{\phi_1, \ldots, \phi_m\}$ is defined as $G(\Phi) = \bigcup_{\phi \in \Phi} G(\{\phi\})$. In general, a domain graph will have both directed and undirected edges.

Barren variables are variables that are neither evidence nor target variables and have only barren descendants, if any (Shachter, 1986). The notion of barren variables can be extended to domain graphs (Madsen, 2006).

The weight $w(X, Y)$ of an edge $(X, Y)$ in a graph $G$ is defined as $w(X, Y) = ||X|| \cdot ||Y||$.

### 2.2 Lazy Propagation

LP computes all single marginals in a BN based on massage passing in a JT $T = (\mathcal{C}, \mathcal{S})$ with cliques $\mathcal{C}$ and separators $\mathcal{S}$. $T$ is constructed from $\mathcal{N} = (\mathcal{X}, G, \mathcal{P})$ by moralisation and triangulation of $G$. Optimal decomposition is, however, NP-hard (Wen, 1991). Hence, the use of heuristics is justified (unless P=NP).

Once $\mathcal{T}$ is constructed, the CPD of each $X \in \mathcal{X}$ is associated with a clique $C$ such that $\text{fa}(X) \subseteq C$, where $\text{fa}(X) = \{X\} \cup \text{pa}(X)$. We let $\Phi_C$ denote the set of CPDs associated with $C \in \mathcal{C}$. As part of the initialisation process CPDs are reduced to reflect the evidence $\epsilon$.

Belief update proceeds by passing messages between cliques over separators in two rounds relative to a root clique. Two messages are passed over each $S \in \mathcal{S}$; one message in each direction. The message $\Phi_{A \to B}$ passed from clique $A$ to clique $B$ consists of a set of probability potentials and it is computed by eliminating variables from a combination of potentials $\Phi_A$ associated with $A$ and messages received from neighboring cliques except $B$ using VE:

$$\Phi_{A \to B} \;=\; \sum_{A \setminus B} (\Phi_A \cup \bigcup_{C \in \text{adj}(A) \setminus \{B\}} \Phi_{C \to A}),$$

where $\text{adj}(A)$ are the cliques adjacent to $C$. Prior to marginalisation, barren variables (and their potentials) are removed, and only potentials for variables not separated from $S$ by $\epsilon$ are included in the calculation of $\Phi_{A \to B}$.

The structure of $T$ induces a partial order on the elimination of $A \setminus B$. This means that for each message $\Phi_{A\to B}$ (or each marginal $P(X \mid \epsilon)$), LP has to determine the elimination order $\sigma$ over $A \setminus B$ on-line. In order not to jeopardise performance, it is important that the algorithm for finding $\sigma$ is fast. As triangulation, in general, is NP-hard, we need to rely on heuristics.

## 3 Variable Elimination Heuristics

In LP, as described above, VE (Zhang and Poole, 1994) (equivalent to the *fusion* operator (Shenoy, 1997) and Bucket elimination (Dechter, 1999)) is used for computing messages and single marginals. If $\Phi$ is a set of potentials and $\mathcal{Y}$ is a set of variables to eliminate from $\Phi$, then LP uses VE to eliminate one variable $Y \in \mathcal{Y}$ at a time by computing:

$$\begin{aligned} \phi_Y &= \sum_Y \prod_{\phi \in \Phi_Y} \phi, \\ \Phi^* &= \Phi \setminus \Phi_Y \cup \{\phi_Y\}, \end{aligned}$$

where $\Phi_Y = \{\phi \in \Phi : Y \in \mathrm{dom}(\phi)\}$.

In the construction of $T$, it may be worth spending additional resources on finding a (near) optimal triangulation as this step is performed only once and any improvement achieved will impact performance of all subsequent belief update operations. On the other hand, for the online triangulation, the aim is to produce an *efficient* elimination order $\sigma$ *fast*. In some cases $\sigma$ has few variables.

For variable $X$, we consider the cost measures $s_d(X)$ (degree of $X$ (or clique candidate size minus one (Rose, 1973))), $s_{dw}(X)$ (sum of the weights of the edges adjacent to $X$), $s_{fi}(X)$ (number of fill-in-edges from eliminating $X$ (Rose, 1973)), $s_{fiw}(X)$ (sum of the weights of the fill-in-edges induced by eliminating $X$ (Jensen, 2012)), $s_{cw}(X)$ (weight of the clique candidate induced by eliminating $X$ (Kjærulff, 1990)) and $s_{H2}(X) = s_{cw}(X)/||X||$ (Cano and Moral, 1994).

Cano and Moral (1994) evaluated six heuristics H1, ..., H6. They found H1 is equivalent to $s_{cw}$, while H3-H6 are variants of $s_{cw}(X)$ adjusted for the size of candidate cliques including $X$, i.e., maximum size or sum of sizes. H2 is as fast to compute as H1 and produces better results, whereas H3 to H6 are expensive to compute. We consider only H2 for online triangulation.

Notice that the scores $s_{dw}(X)$, $s_{fiw}(X)$, $s_{cw}(X)$ and $s_{H2}(X)$ all use the notion of *weight* of an edge $(X, Y)$ between $X$ and an adjacent variable $Y$ or a set of variables.

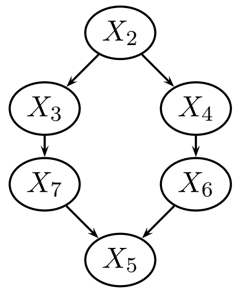


Figure 1: The domain graph for Example 1.

**Example 1.** Let $\Phi = \{\phi(X_2), \phi(X_3 \mid X_2), \phi(X_4 \mid X_2), \phi(X_7 \mid X_3), \phi(X_6 \mid X_4), \phi(X_5 \mid X_7, X_8)\}$ be a set of potentials, with $G(\Phi)$ shown in Figure 1, from which we want to compute $\phi(X_5)$ and assume $||X_i|| = i$. Different heuristics will produce different elimination orders for the computation of $\phi(X_5)$ and assign the same score to some variables. For instance, $s_{fi}(X_2) = s_{fi}(X_3) = s_{fi}(X_4) = 1$, $s_{fiw}(X_2) = s_{fiw}(X_4) = s_{H2}(X_2) = s_{H2}(X_4) = 12$, and $s_{dw}(X_2) = 14$.

Example 1 also illustrates that none of the heuristics finds the optimal order $\sigma = (X_4, X_3, X_2, X_6, X_7)$.

## 4 The *VarElim* Algorithm

*VarElim* is a new algorithm for eliminating a set of variables $\mathcal{Y}$ from a set of potentials $\Phi$ with domain graph $G(\Phi)$, taking as input a sequence of cost measures $\mathcal{S} = (s_1, \ldots, s_n)$ to identify the next variable to eliminate (see Algorithm 1). Starting with $i = 1$, if there is a tie for the best value in a cost measure $s_i$, the algorithm proceeds to consider $s_{i+1}$. The algorithm proceeds through the order to identify the variable to eliminate next.

**Data**: Let $\mathcal{Y}$ be the variables to eliminate from $\Phi$ and $\mathcal{S} = (s_1, \ldots, s_n)$ be an ordered sequence of score functions.
**Result**: $\Phi^* = \sum_{\mathcal{Y}} \Phi$.
**begin**
1 **foreach** $Y \in \mathcal{Y}$ **do**
  **for** $i = 1$ **to** $n$ **do** Compute $s_i(Y)$
  **end**
  $\mathcal{A}_0 = \mathcal{Y}$
2 **while** $|\mathcal{A}_0| > 0$ **do**
  **for** $i = 1$ **to** $n$ **do**
3   Set $\mathcal{A}_i = \arg_{Y \in \mathcal{A}_{i-1}} \min s_i(Y)$
  **end**
4   Eliminate $Y \in \mathcal{A}_n$
  Set $\mathcal{A}_0 = \mathcal{A}_0 \setminus \{Y\}$
5   Recompute $s_i(Y')$ for $Y' \in \text{adj}(Y)$, for $i = 1, \ldots, n$
  **end**
**end**

**Algorithm 1:** *VarElim.*

The loop in Step 1 computes the costs $s_i(Y)$ for $Y \in \mathcal{Y}$. The loop in Step 2 iterates until all variables are eliminated. In Step 3, $\mathcal{A}_i \subseteq \mathcal{A}_{i-1}$ is the set of variables with minimum cost in the score function $s_i$. Notice that $|\mathcal{A}_{i-1}| = 1$ means that $Y$ is unique. If $|\mathcal{A}_n| = 1$, then a unique variable has been identified. If $|\mathcal{A}_n| > 1$, then these variables received the same score for each $s_i$, for $i = 1, 2, \ldots, n$ and $Y$ is selected at random in Step 4. In this case, the score functions in $\mathcal{S}$ were not able to break the ties between elements of $\mathcal{A}_n$. In Step 4, $Y$ is eliminated as explained in Section 3 producing an updated set of potentials $\Phi^*$. Notice that $\Phi$ is updated at each iteration and $s_i$ is recomputed in Step 5 from $G(\Phi^*)$.

Breaking ties in cost measures in relation to LP with AR was studied by Butz et al. (2011) as well as Madsen and Butz (2012), whereas Cano and Moral (1994) suggested tie breaking rather than selecting randomly between variables with equally good scores, and Velev and Gao (2009) suggested tie breaking by looking at the degrees of variables adjacent to the variable to eliminate.

Table 1: BNs and JTs.

| $\mathcal{N}$ | $\lvert\mathcal{X}\rvert$, $\lvert\mathcal{C}\rvert$ | $\max_{C \in \mathcal{C}} s(C)$ | $\sum_{C \in \mathcal{C}} s(C)$ |
|---|---|---|---|
| Barley | 48, 36 | 7,257,600 | 17,140,796 |
| KK | 50, 38 | 5,806,080 | 14,011,466 |
| Mildew | 35, 29 | 1,249,280 | 3,400,464 |
| OOW_solo | 40, 29 | 1,644,300 | 4,651,788 |
| ship-ship | 50, 35 | 4,032,000 | 24,258,572 |

Bertele and Brioschi (1972) considered the sequences ($d$, *fi*) and (*fi*, $d$) to break ties.

**Example 2.** Consider again Example 1 computing $\phi(X_5)$ using *VarElim* with $\mathcal{S} = (s_{fi}, s_{fiw}, s_{H2}, s_{dw})$. The iteration in Step 3 will produce $\mathcal{A}_1 = \{X_2, X_3, X_4, X_6, X_7\}$, $\mathcal{A}_1 = \{X_2, X_3, X_4\}$, $\mathcal{A}_2 = \{X_2, X_4\}$, $\mathcal{A}_3 = \{X_2, X_4\}$ and $\mathcal{A}_4 = \{X_2\}$. This means that $X_2$ is selected as the first variable to eliminate. The algorithm continues in the order $\sigma = (X_3, X_4, X_6, X_7)$.

Notice that if two consecutive heuristics $h_i$ and $h_{i+1}$ in $\mathcal{S}$ always produce the same number of ties, this means that one of $h_i$ and $h_{i+1}$ is redundant and does not improve the tie breaking.

## 5 Empirical Evaluation

### 5.1 Experimental Setup

The experiments were performed using both real-world and randomly generated networks. We report only the results for the five real-world networks (Madsen, 2010) in Table 1, where $s(C) = \prod_{X \in C} |\text{dom}(X)|$. JTs have been generated using the *total weight* heuristic (Jensen, 2012) who cites (Shoikhet and Geiger, 1997). All single marginals are computed for ten different sets of evidence, for each $|\mathcal{X}_\epsilon| = 0, \ldots, n$, where $n = |\mathcal{X}|$.

In the experiments, a greedy variant of *VarElim* is used. Instead of generating all candidates with the same score in each iteration, it keeps track of the best scoring variable and use the score sequence to compare and select variables.

The experiments were performed using a Java implementation (Java (TM) 2 Runtime Environment, Standard Edition (build 1.5.0_22-b03)) running on a Linux Ubuntu (kernel 2.6.38-11-server) server with an Intel Xeon(TM) E3-1270 Processor (3.4GHz, 4C/8T and 8MB

Cache) and 32 GB RAM.

### 5.2 Different Heuristics

The aim is to analyse the impact of the elimination heuristics of Section 3 on the performance of LP. This corresponds to calling *VarElim* with $|\mathcal{S}| = 1$. We compare *VarElim* using min and max in Step 3 to assess the robustness of LP with respect to each cost measure.

Figure 2 shows the performance of LP using $s_{dw}$ in Barley. The difference in performance between min and max is most significant for small $|\mathcal{X}_\epsilon|$ and decreases as $|\mathcal{X}_\epsilon|$ increases.

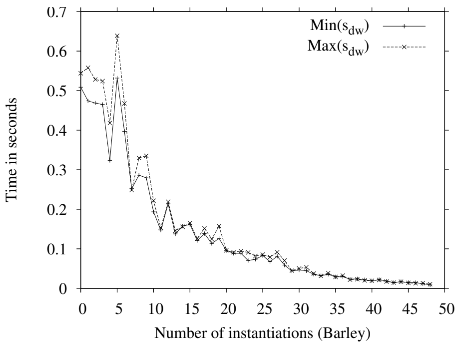


Figure 2: Belief update in Barley using $s_{dw}$.

Figures 3-5 show the performance of LP using *fiw*, *fi* and *H2* in ship-ship, respectively. The performance difference between *fiw*, *fi* and *H2* is insignificant for the min versions. This is the case across many examples considered in the tests.

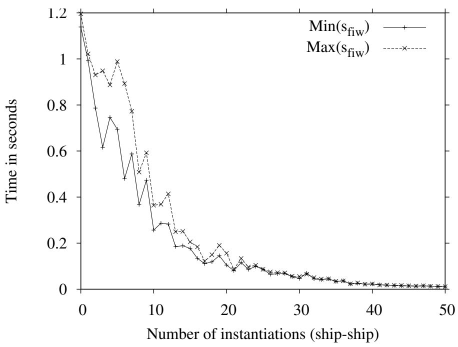


Figure 3: Belief update in ship-ship using *fiw*.

LP appears to be robust with respect to the heuristic used for the networks and JTs considered in the evaluation. It is important to reiterate that the JTs have been generated using

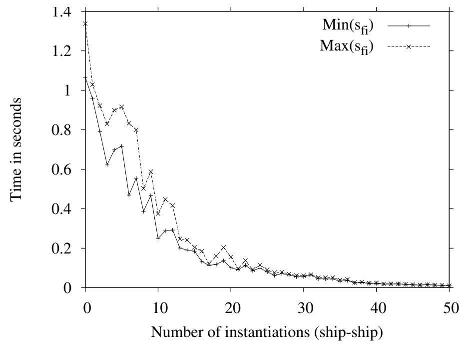


Figure 4: Belief update in ship-ship using *fi*.

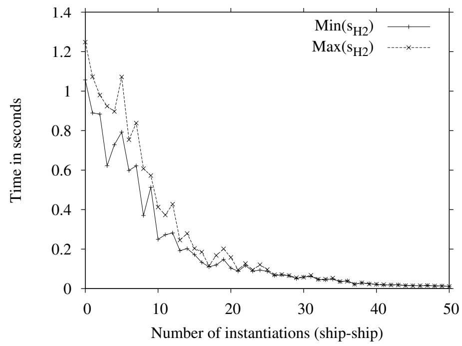


Figure 5: Belief update in ship-ship using *H2*.

the *total weight* heuristic that is known to often produce near optimal triangulations (Jensen, 2012). This may have a significant impact on the results.

### 5.3 Minimum or Maximum Costs

Based on the results reported in the previous section and by Kjærulff (1990), we consider four sequences of cost measures $\mathcal{S}_0 = (s_{fiw}, s_{cw}, s_{dw}, s_{fi}, s_d)$ and its reverse sequence $\mathcal{S}_1 = (s_d, s_{fi}, s_{dw}, s_{cw}, s_{fiw})$ as well as $\mathcal{S}_2 = (s_{dw}, s_{fiw}, s_{cw}, s_d, s_{fi})$ and $\mathcal{S}_3 = (s_{H2}, s_{fiw}, s_{cw}, s_{dw}, s_{fi})$.

The aim is to analyse the impact cost measure sequences can have on performance and to assess if the potential performance improvement is robust with respect to the sequence in which the cost measures are applied. The experimental results reported in this section are focused on a comparison of *VarElim* using min in Step 3 with *VarElim* using max in Step 3.

Figure 6 shows the performance in ship-ship using $\mathcal{S}_0$ (performance is similar for $\mathcal{S}_1$-$\mathcal{S}_3$), and

Figure 7 shows the performance in KK using $\mathcal{S}_1$ (performance is similar for $\mathcal{S}_0$, $\mathcal{S}_2$ and $\mathcal{S}_3$).

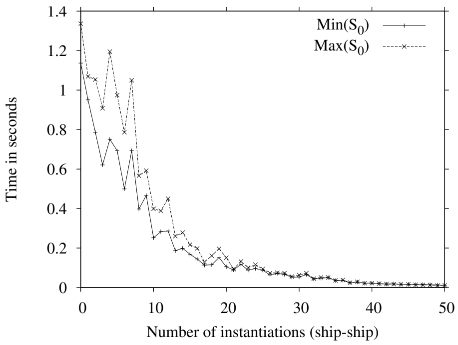


Figure 6: Belief update in ship-ship using $\mathcal{S}_0$.

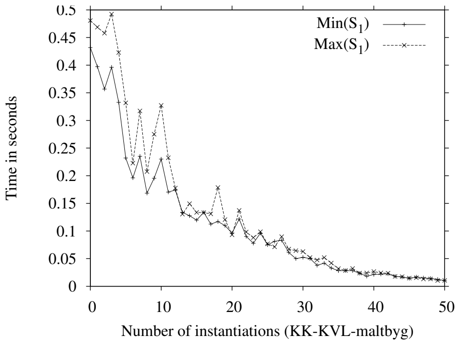


Figure 7: Belief update in KK using $\mathcal{S}_1$.

Figure 6 and Figure 7 illustrate that the difference in performance between using min and max in Step 3 can be significant for the orders generated. The results suggest that the partial order induced by $T$ ensures that performance is not extremely jeopardised by using max.

### 5.4 Best and Worst Tie Breaking

The purpose of this experiment is to analyse the potential impact cost measure sequences can have on performance, once there is a tie for minimum value in the first cost measure used, and to assess if the potential performance improvement is robust with respect to the sequence in which the cost measures are applied.

The experiment compares *VarElim* using min in Step 3 with *VarElim* using min in Step 3 on the first iteration and max in subsequent iterations. Thus, the aim is to analyse the impact of best and worst possible tie breaking once the initial score is selected using min. Notice that best tie breaking method is equivalent to the minimum costs method.

Figure 8 shows the performance on OOW_solo using $\mathcal{S}_0$. The performance is similar for $\mathcal{S}_1$-$\mathcal{S}_3$. In fact, for most networks considered in the experiments, the difference is insignificant and can be both positive and negative.

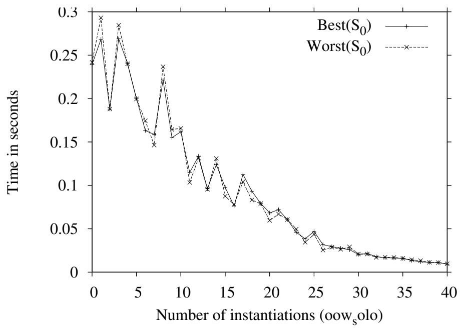


Figure 8: Belief update in OOW_solo using $\mathcal{S}_0$.

Figure 9 shows the number of ties encountered for each $s_i$ ($i = 1, \ldots, 5$) in $\mathcal{S}_0$ on OOW_solo. It also shows that the number of ties decreases as *VarElim* iterates through the scores and that for this network subsequences $(s_{cw}, s_{dw})$ and $(s_{fi}, s_d)$ are not effective.

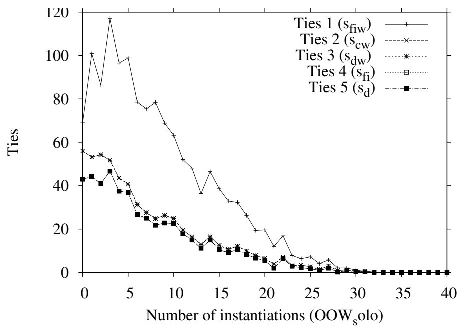


Figure 9: Score ties, OOW_solo using $\mathcal{S}_0$.

In the experiments *total weight* triangulations (often optimal in clique weight) have been used for constructing $T$. To investigate the impact of optimal or what is believed to be near optimal triangulations, non-optimal JTs (referred to as $T_{cs}$) using the minimum clique size heuristic are generated (see Table 2).

Figure 10 shows the performance in Barley using $\mathcal{S}_0$ on $T_{cs}$ (for $\mathcal{S}_1$ - $\mathcal{S}_3$ there is almost no

Table 2: Minimum clique size JTs.

| $\mathcal{N}$ | $\lvert\mathcal{C}\rvert$ | $\max_{C \in \mathcal{C}} s(C)$ | $\sum_{C \in \mathcal{C}} s(C)$ |
|---|---|---|---|
| Barley | 36 | 13,063,680 | 24,970,779 |
| Mildew | 29 | 1,756,800 | 4,686,212 |
| OOW_solo | 29 | 17,010,000 | 32,383,477 |

difference). Figure 11 shows the performance in Mildew using $\mathcal{S}_1$ on $T_{cs}$ (for $\mathcal{S}_0$, $\mathcal{S}_2$ and $\mathcal{S}_3$ results are similar).

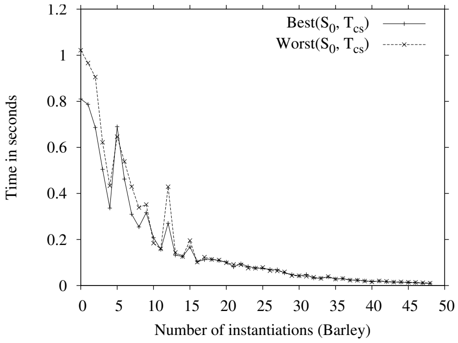


Figure 10: Barley on $T_{cs}$ using $\mathcal{S}_0$.

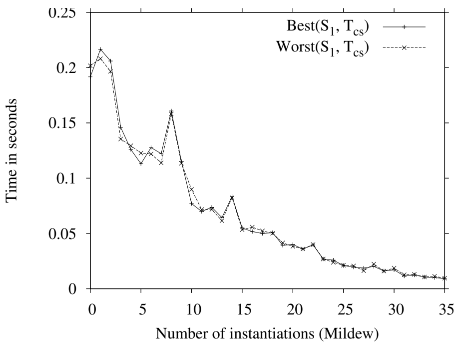


Figure 11: Mildew on $T_{cs}$ using $\mathcal{S}_1$.

Figure 12 shows the performance in OOW_solo using $\mathcal{S}_1$ on $T_{cs}$ (results for $\mathcal{S}_0$ are similar). For $\mathcal{S}_2$ and $\mathcal{S}_3$ there is no difference and performance is at the level of $\mathcal{S}_1$ best. As expected, LP has higher cost on $T_{cs}$ than on $T$. This suggests that the initial partial order of the JT is more important for performance than the orders identified online.

## 6 Discussion and Conclusion

This paper has considered the importance of elimination heuristics in LP when using VE as

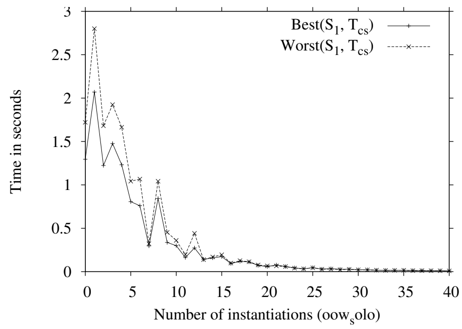


Figure 12: OOW_solo on $T_{cs}$ using $\mathcal{S}_1$.

the message and single marginal computation algorithm. The paper has introduced *VarElim* as an algorithm for eliminating a set of variables from a set of potentials using a sequence of cost measures, as opposed to using only a single cost measure. The sequence of cost measures is used for score tie breaking when identifying the variable to eliminate next.

The paper reports on a number of empirical evaluations on the performance of LP with respect to the elimination heuristics used to identify the next variable to eliminate. The results indicate that LP is robust relative to the cost measures used. This indicates that the structure of the JT is more important for performance than the online elimination heuristic. The experiments with min and max show that the time cost is increased by consistently selecting the highest scoring variable to eliminate next. The experiments with best and worst tie breaking show that breaking ties does not have a significant impact on performance when the initial JT is optimal (or believed to be near optimal), whereas the impact is more significant for less optimal JTs.

**Acknowledgments**

This research was partly supported by the Spanish Ministry of Economy and Competitiveness under project TIN2010-20900-C04-01 and the European Regional Development Fund (FEDER).

## References

U. Bertele and F. Brioschi. 1972. *Nonserial Dynamic Programming.* Academic Press.

M. Bloemeke and M. Valtorta. 1998. A Hybrid Algorithm to Compute Marginal and Joint Beliefs in Bayesian Networks and its Complexity. In *Proc. of UAI*, pages 416–23.

C. J. Butz, A. L. Madsen, and K. Williams. 2011. Using four cost measures to determine arc reversal orderings. In *Proc. of ECSQARU*, pages 110–121.

C. Cannings, E. A. Thompson, and H. H. Skolnick. 1978. Probability functions on complex pedigrees. *Advances in Applied Probability*, 10:26–61.

A. Cano and S. Moral. 1994. Heuristic algorithms for the triangulation of graphs. In *Proc. of IPMU*, pages 166–171.

G. F. Cooper. 1990a. Bayesian Belief-Network Inference Using Recursive Decomposition. Technical Report KSL 90-05, Knowledge Systems Laboratory.

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

R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter. 1999. *Probabilistic Networks and Expert Systems*. Springer.

P. Dagum and M. Luby. 1993. Approximating probabilistic inference in Bayesian belief netwoks is NP-hard. *Artificial Intelligence*, 60:141–153.

R. Dechter. 1999. Bucket elimination: A unifying framework for probabilistic inference. *Artificial Intelligence*, 113(1-2):41–85.

F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. 1990. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, 4:269–282.

F. Jensen. 2012. HUGIN API Reference Manual. www.hugin.com. Version 7.6.

J. H. Kim and J. Pearl. 1983. A computational model for causal and diagnostic reasoning in inference systems. In *Proc. of IJCAI*, pages 190–193.

U. B. Kjærulff and A. L. Madsen. 2008. *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer.

U. B. Kjærulff. 1990. Graph triangulation — algorithms giving small total state space. Technical Report R 90-09, University of Aalborg, Denmark.

S. L. Lauritzen and D. J. Spiegelhalter. 1988. Local computations with probabilities on graphical structures and their application to expert systems. *JRSS, B*, 50(2):157–224.

Z. Li and B. D'Ambrosio. 1994. Efficient Inference in Bayes Networks As A Combinatorial Optimization Problem. *IJAR*, 11(1):55–81.

A. L. Madsen and C. J. Butz. 2012. Ordering arc-reversal operations when eliminating variables in lazy ar propagation. Working paper, Aalborg University.

A. L. Madsen and F. V. Jensen. 1999. Lazy propagation: A junction tree inference algorithm based on lazy evaluation. *Artificial Intelligence*, 113(1–2):203–245.

A. L. Madsen. 2006. Variations Over the Message Computation Algorithm of Lazy Propagation. *IEEE Transactions on Systems, Man. and Cybernetics Part B*, 36(3):636–648.

A. L. Madsen. 2010. Improvements to Message Computation in Lazy Propagation. *IJAR*, 51(5):499–514.

S. M. Olmsted. 1983. *On representing and solving decision problems*. PhD thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.

J. Pearl. 1982. Reverend Bayes on inference engines: A distributed hierarchical approach. In *Proc. of AAAI*, pages 133–136.

J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Series in Representation and Reasoning. Morgan Kaufmann Publishers, San Mateo, CA.

D. J. Rose. 1973. A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations. In *Graph Theory and Computing*, pages 183–217.

R. Shachter, B. D'Ambrosio, and B. Del Favero. 1990. Symbolic probabilistic inference in belief networks. In *Proc. of National Conference on AI*, pages 126–131.

R. D. Shachter. 1986. Evaluating influence diagrams. *Operations Research*, 34(6):871–882.

G. R. Shafer. 1996. *Probabilistic Expert Systems*. SIAM.

P. P. Shenoy and G. Shafer. 1990. Axioms for probability and belief-function propagation. In *Proc. of UAI*, pages 169–198.

P. P. Shenoy. 1997. Binary join trees for computing marginals in the Shenoy-Shafer architecture. *IJAR*, 17(2-3):239–263.

K Shoikhet and D Geiger. 1997. A practical algorithm for finding optimal triangulations. In *AAAI/IAAI'97*, pages 185–190.

M. N. Velev and P. Gao. 2009. Efficient sat techniques for absolute encoding of permutation problems: Application to hamiltonian cycles. In *Proc. of ISQED*, pages 371–376.

W. X. Wen. 1991. Optimal decomposition of belief networks. In *Proc. of UAI*, pages 209–224.

N. L. Zhang and D. Poole. 1994. A simple approach to Bayesian network computations. In *Proc. Canadian Conference on AI*, pages 171–178.