International Journal of Approximate Reasoning 54 (2013) 1182–1196

Contents lists available at SciVerse ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Ordering arc-reversal operations when eliminating variables in lazy AR propagation

Anders L. Madsen [a,b,*], Cory J. Butz [c]

[a] HUGIN EXPERT A/S, Gasværksvej 5, DK-9000 Aalborg, Denmark
[b] Department of Computer Science, Aalborg University, Selma Lagerlöfs Vej 300, DK-9220 Aalborg Ø, Denmark
[c] Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada S4S 0A2

ARTICLE INFO

Article history:
Received 15 May 2012
Received in revised form 15 February 2013
Accepted 15 February 2013
Available online 4 March 2013

Keywords:
Bayesian networks
Lazy AR propagation
Arc-reversal cost measures

ABSTRACT

This paper considers the problem of ordering arc-reversal operations and breaking ties in cost measures when eliminating variables in *Lazy AR Propagation* (LPAR). In particular, the paper presents the *BreakTies* algorithm for breaking ties in cost measures when selecting the next arc to reverse in a variable elimination operation. *BreakTies* is based upon using a sequence of cost measures instead of randomly selecting an arc to reverse when multiple arcs share the same cost. The paper reports on an experimental evaluation of LPAR for belief update in Bayesian networks considering six sequences of five cost measures for breaking ties using *BreakTies*. The experimental results show that using *BreakTies* to select the next arc to reverse in a variable elimination operation can improve performance of LPAR.

1. Introduction

Bayesian networks [24,6,13,14,7,15] provide a rigorous foundation for uncertainty management by combining probability theory and graph theory. They have been applied in practice to a range of different problem domains, including finance, food safety, agriculture and bioinformatics [25,9,22]. A Bayesian network consists of a *directed acyclic graph* (DAG) and a corresponding set of *conditional probability tables* (CPTs). The vertices in the DAG represent the random variables in the real-world problem, while the arcs in the graph represent probabilistic dependencies amongst the variables. More specifically, the probabilistic conditional independencies encoded in the DAG define the product of the given CPTs as a joint probability distribution.

While [5] has shown that the complexity of exact inference in discrete Bayesian networks is NP-hard, several approaches have been put forth that work quite well in practice. *Arc-reversal* (AR) [23,27] eliminates a variable by reversing the arcs between the variable and its children and then building the CPTs corresponding to the modified DAG. Another approach, called *variable elimination* (VE) [33,1], eliminates a variable by multiplying together all of the distributions involving the variable and then summing the variable out of the obtained product. VE is similar to the Fusion operator [29] and Bucket elimination [8]. Yet another technique, called *Symbolic Probabilistic Inference* (SPI) [26,17], focuses on the order in which distributions are combined and eliminates variables when possible. We refer to the above methods as emphasize term, since they manipulate the set of CPTs associated with the Bayesian network directly. On the other hand, *junction tree propagation*, which [28] emphasizes is central to the theory and practice of probabilistic expert systems, first builds a secondary network, called a junction tree, from the DAG of the Bayesian network and then performs inference by propagating probabilities in the junction tree [16,12,30].

* Corresponding author at: HUGIN EXPERT A/S, Gasværksvej 5, DK-9000 Aalborg, Denmark.
E-mail addresses: alm@hugin.dk (A.L. Madsen), butz@cs.uregina.ca (C.J. Butz).

0888-613X/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ijar.2013.02.007

Lazy Propagation (LP) [21] is a hybrid inference algorithm, combining junction tree propagation and direct methods for computing all posterior marginals. LP is different, particular in that it maintains a factorization of distributions allowing for exploitation of *barren* variables [27] and independencies induced by evidence [16,10]. Consequently, LP often outperforms the traditional junction tree propagation algorithms. LP performs message passing based on the scheme of Shenoy–Shafer propagation in a junction tree, while messages are computed by VE. Madsen [18,19] modified LP by replacing VE as the message construction algorithm with AR or SPI. Madsen called the former setting LPAR and conducted an empirical study of LP and LPAR. Butz et al. [2] introduced a junction tree algorithm that selectively applies either AR or VE to build the propagated messages, while Cinicioglu and Shenoy [4] discussed arc reversals in hybrid Bayesian networks with deterministic variables.

Madsen [20] demonstrated that the order in which arcs are reversed in LPAR can affect the amount of computation required during belief update. Preliminary experimental results have suggested that *cpt-weight* (*cptw*) is the best of the four measures considered [20]. Butz et al. [3] introduced the *BreakTies* algorithm for using multiple cost measures to order arc-reversal operations when eliminating a variable in LPAR. This work was motivated by the experimental observation that ties in a cost measure occur quite often during inference. The example in Section 3 illustrates how this approach can reduce the number of calculations performed compared to the approach in [20], which reverses the first arc tied for the best score.

In this paper, we introduce a new cost measure, investigate the impact of different cost measure sequences on belief update performance, and extend the empirical evaluation significantly. The empirical evaluation was performed on a set of 25 real-world networks. The score *dw* is proposed to measure the sum of parent and child edge weights before arc-reversal, a cost not considered in [3]. Experiments selecting the next arc to reverse based on selecting the maximum *dw* score versus the minimum *dw* reveal a significant potential of this score. Furthermore, the *dw* cost measure is compared against four other cost measures considered in this paper. Next, we consider six different cost measure sequences developed based on previous results and the results of the evaluation on single cost measures. The experimental results demonstrate a significant difference in average time cost between *best* tie breaking and selecting the maximum cost arc to reverse when using the *BreakTies* algorithm. The results also show that *best* tie breaking almost always reduces the average time cost of belief update over both *worst* tie breaking and using only the first cost measure in the sequence to select the arc to reverse next. The extensive results in Section 4 demonstrate a small but observable computational improvement.

The remainder of the paper is organized as follows. Section 2 contains preliminaries and introduces the notation used. In Section 3, we introduce our approach based on the *BreakTies* algorithm for using multiple cost measures in ordering arc-reversal operations. Experimental findings are given in Section 4. Conclusions are drawn in Section 5.

2. Preliminaries and notation

In this section, results from Bayesian networks, AR, cost measures, and LPAR, are reviewed.

2.1. Bayesian networks

Consider a finite set of discrete random variables $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$. Let $\mathrm{dom}(X_i)$ denote the finite domain of values that each variable $X_i \in \mathcal{X}$ can assume. For a subset $\mathcal{Y} \subseteq \mathcal{X}$, the Cartesian product of the domains of the individual variables in $\mathcal{Y}$ is denoted $\mathrm{dom}(\mathcal{Y})$. An element $y \in \mathrm{dom}(\mathcal{Y})$ is a *configuration* or *row* of $\mathcal{Y}$. A *potential* [28] on $\mathrm{dom}(\mathcal{Y})$ is a function ϕ such that $\phi(x) \geq 0$, for each configuration $y \in \mathrm{dom}(\mathcal{Y})$, and at least one $\phi(y)$ is positive. For simplicity we speak of a potential as defined on $\mathcal{Y}$ instead of on $\mathrm{dom}(\mathcal{Y})$, and we call $\mathcal{Y}$ its domain rather than $\mathrm{dom}(\mathcal{Y})$ [28]. A *joint probability distribution* [28] on $\mathcal{X}$, written $p(\mathcal{X})$, is a function p on $\mathcal{X}$ satisfying the following two conditions: (i) $0 \leq p(x) \leq 1$, for each configuration $x \in \mathrm{dom}(\mathcal{X})$; (ii) $\sum_{x \in \mathrm{dom}(\mathcal{X})} p(x) = 1$. Let $\mathcal{Y}_i$ and $\mathcal{Y}_j$ be two disjoint subsets of $\mathcal{X}$. A *conditional probability table* (CPT) for $\mathcal{Y}_i$ given $\mathcal{Y}_j$, denoted $p(\mathcal{Y}_i|\mathcal{Y}_j)$, is a nonnegative function on $\mathcal{Y}_i \cup \mathcal{Y}_j$, satisfying the following condition: for each configuration $y_j \in \mathrm{dom}(\mathcal{Y}_j)$, $\sum_{y_i \in \mathrm{dom}(\mathcal{Y}_i)} p(\mathcal{Y}_i = y_i \mid \mathcal{Y}_j = y_j) = 1$.

A discrete *Bayesian network* $\mathcal{N}$ over $\mathcal{X} = \{X_1, X_2, \ldots, X_n\}$ is a triple $\mathcal{N} = (\mathcal{X}, G, \mathcal{P})$. Here G is a DAG with vertex set V corresponding one-to-one with $\mathcal{X}$ and edge set E, and $\mathcal{P}$ is the set of CPTs $\{p(X_i|P_i) \ : \ i = 1, 2, \ldots, n\}$, where P_i denotes the parents of variable $X_i \in D$. For example, one Bayesian network is the DAG in Fig. 1 (i) together with CPTs $p(X_1)$, $p(X_2)$, $p(X_3|X_1)$ and $p(X_4|X_1, X_2)$. Here, the parents P_4 of variable X_4 are $P_4 = \{X_1, X_2\}$.

The *family* F_i of a variable X_i in a Bayesian network is the variable together with its parents, that is, $F_i = \{X_i\} \cup P_i$. We let $ch(X_i)$ denote the *children* of X_i, that is, $\{X_l : X_i \in P_l\}$. By $p(\mathcal{Y}_i|\mathcal{Y}_j)$, we always mean $p(\mathcal{Y}_i|\mathcal{Y}_j - \mathcal{Y}_i)$. We use WZ to mean $W \cup Z$.

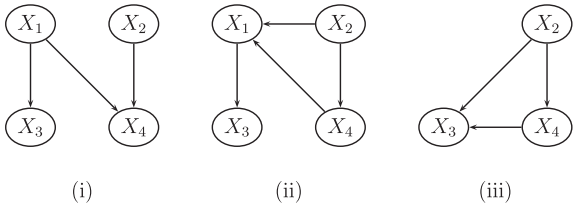

Fig. 1. Eliminating X_1 in (i) by reversing arc (X_1, X_4) (ii) followed by arc (X_1, X_3) (iii).

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 ϵ, where $\mathcal{X}_\epsilon \subseteq \mathcal{X}$ is the set of variables instantiated by ϵ.

2.2. Arc-reversal

AR eliminates a variable X_i by reversing the arcs (X_i, X_j) for each child X_j of X_i, where $j = 1, 2, \ldots, k$. With respect to multiplication, addition, and division, AR reverses one arc (X_i, X_j) as a three-step process:

$$p(X_i, X_j|P_iP_j) = p(X_i|P_i) \cdot p(X_j|P_j), \tag{1}$$

$$p(X_j|P_iP_j - \{X_i\}) = \sum_{X_i} p(X_i, X_j|P_iP_j), \tag{2}$$

$$p(X_i|P_iP_jX_j) = \frac{p(X_i, X_j|P_iP_j)}{p(X_j|P_iP_j - \{X_i\})}. \tag{3}$$

Suppose the variable X_i to be eliminated has k children. The distributions defined in (1) - (3) are built for the first $k - 1$ children. For the last child X_k, however, only the distributions in (1) - (2) are built. When considering X_k, there is no need to build the final distribution for X_i in (3), since X_i will be eliminated as a barren variable in the next step of the process. Therefore, AR eliminates a variable X_i with k children by building $3k - 1$ distributions. However, AR only outputs the k distributions built in (2).

2.3. Cost measures for ordering AR operations

Madsen [20] considers four different cost measures for ordering AR operations in the elimination of a variable in LPAR. The elimination of variable X_i by a sequence of AR operations produces an ordering of the X_i's children $\mathrm{ch}(X_i)$. We call this ordering a *child ordering* and denote it by ρ. The child ordering ρ determines the set of induced edges, which have an impact on the performance of belief update. Thereby, the child ordering $\hat{\rho}$ leading to the best time and space performance should be chosen. Since it is not possible by local computations only to identify the ordering $\hat{\rho}$ having the best time and space cost, the focus is on identifying a ordering ρ with minimum local cost. Notice that the elimination of a variable by arc reversal may introduced new edges to be reversed subsequently.

Five different myopic score functions for determining the cost of reversing an arc are considered: *cptw* (s_{cptw}), *fiw* (s_{fiw}), *fi* (s_{fi}), *nop* (s_{nop}), and *dw* (s_{dw}). The objective is to determine a child ordering ρ with as low a cost (in the score function used) as possible. We conjecture that the problem of identifying the optimal cost (in the score function used) is NP-hard as the problem is similar to identifying the optimal decomposition [31,32].

The cost of reversing arc (X_i, X_j) using the *cptw* score s_{cptw} is:

$$s_{cptw}(X_i, X_j) = \prod_{X_k \in F_iF_j} |\mathrm{dom}(X_k)|.$$

Thus, *cptw* is defined as the total state space size of X_i's CPT after reversal.

Using the *fiw* score s_{fiw}, the cost of reversing (X_i, X_j) is:

$$s_{fiw}(X_i, X_j) = \sum_{X_k \in F_j - F_i} w(X_k, X_i) + \sum_{X_l \in P_i - P_j} w(X_l, X_j),$$

where $w(X_a, X_b) = |\mathrm{dom}(X_a)| \cdot |\mathrm{dom}(X_b)|$. Then, *fiw* cost is equal to the sum of the edge weights of the new parents $F_j - F_i$ of X_i and $P_i - P_j$ of X_j.

The cost of reversing arc (X_i, X_j) using the *fi* score s_{fi} is:

$$s_{fi}(X_i, X_j) = |F_j - F_i| + |P_i - P_j|,$$

namely, the *fi* cost is equal to the number of new parents $F_j - F_i$ of X_i and new parents $P_i - P_j$ of X_j.

Using the *nop* score s_{nop}, the cost of reversing (X_i, X_j) is:

$$s_{nop}(X_i, X_j) = |P_i \cup F_j|,$$

i.e., the *nop* cost is the cardinality of X_i's parents after reversing (X_i, X_j).

A preliminary experimental analysis suggests an effectiveness ranking of s_{cptw}, s_{fiw}, s_{fi}, and s_{nop} [20] for the scores.

Using the new score *dw*, denoted s_{dw}, the cost of reversing (X_i, X_j) is:

$$s_{dw}(X_i, X_j) = \sum_{X_k \in \mathrm{ch}(X_j)} w(X_j, X_k) + \sum_{X_l \in P_j} w(X_l, X_j),$$

i.e., the *dw* cost is equal to the sum of the edge weights of parents and children of X_j before reversing (X_i, X_j).

In Section 4, we report on an empirical analysis of the effectiveness of cost measure sequences for breaking ties.

Example 1. Consider eliminating variable X_1 in Fig. 1(i), where X_1, X_2, X_4 are binary and X_3's domain has four values. Assume we want to select the next arc to reverse using s_{cptw}. Then, we compute the *cptw* score of arcs $\mathcal{A}$ corresponding to the children X_3 and X_4 of X_1:

$$s_{cptw}(X_1, X_3) = \prod_{X_k \in \{X_1, X_3\}} |\mathrm{dom}(X_k)| = 2 \cdot 4 = 8,$$
$$s_{cptw}(X_1, X_4) = \prod_{X_k \in \{X_1, X_2, X_4\}} |\mathrm{dom}(X_k)| = 2 \cdot 2 \cdot 2 = 8.$$

Since $s_{cptw}(X_1, X_3)$ is equal to $s_{cptw}(X_1, X_4)$, *cptw* does not distinguish between arcs (X_1, X_3) and (X_1, X_4). Thus, one arc is randomly chosen, say (X_1, X_4), and reversed:

$$p(X_1, X_4|X_2) = p(X_1) \cdot p(X_4|X_1, X_2),$$
$$p(X_4|X_2) = \sum_{X_1} p(X_1, X_4|X_2),$$
$$p(X_1|X_2, X_4) = \frac{p(X_1, X_4|X_2)}{p(X_4|X_2)}.$$

The resulting DAG is shown in Fig. 1(ii). The reversal of the other arc (X_1, X_3) gives Fig. 1(iii) by computing:

$$p(X_1, X_3|X_2, X_4) = p(X_1|X_2, X_4) \cdot p(X_3|X_1),$$
$$p(X_3|X_2, X_4) = \sum_{X_1} p(X_1, X_3|X_2, X_4).$$

2.4. Lazy AR propagation

LP [21] is a method for computing all marginals in a Bayesian network based on message passing in a corresponding junction tree. LP combines message passing in a junction tree with direct methods for computation of messages and marginals with the aim of taking advantage of the structure of potentials and distributions to exploit independence and irrelevance properties induced by the evidence being propagated.

LP is performed efficiently in a junction tree structure created from the DAG G of the Bayesian network $\mathcal{N} = (\mathcal{X}, G, \mathcal{P})$. We abstain here from explaining all the details in the compilation procedure. In brevity, the compilation of $\mathcal{N}$ into a junction tree $\mathcal{T} = (\mathcal{C}, \mathcal{S})$ with cliques $\mathcal{C}$ and separators $\mathcal{S}$ consists of three steps. The first step constructs the moral graph G^m of G by adding an edge between each pair of unconnected vertices with a common child and removing direction on the edges. The second step builds a triangulated graph G^T from G^m by adding undirected edges until every cycle of length greater than three has a chord. The third and final step annotates the cliques $\mathcal{C}$ and the separators $\mathcal{S}$ as the nodes and links of $\mathcal{T}$, respectively. Each clique $C \in \mathcal{C}$ represents a maximal complete sub-graph in the triangulated graph G^T.

Once $\mathcal{T}$ is constructed the CPT of each $X_i \in \mathcal{X}$ is associated with a clique C such that $F_i \subseteq C$. We let Φ_C denote the set of CPTs associated with $C \in \mathcal{C}$. Notice that CPTs associated with a clique are not combined. As part of the initialisation process, CPTs are instantiated to reflect the evidence ϵ.

Belief update proceeds as a two-phase process, where information is passed as messages between cliques over separators. 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 Φ_A associated with A and messages received from neighbouring cliques except B:

$$\Phi_{A\to B} = \left(\Phi_A \cup \bigcup_{C\in \mathrm{adj}(A)\setminus\{B\}} \Phi_{C\to A}\right)^{M\downarrow B},$$

where the marginalisation operation M is AR in LPAR and adj(A) are the cliques adjacent to C. The notation $M \downarrow B$ specifies that the set of potentials are marginalised to B using M as the marginalization operator. When using AR as the marginalisation operation it is necessary to avoid creating directed cycles in the directed graphs induced by the set of probability potentials being marginalised. Potentials for which all head variables are barren and potentials over variables which are all separated from B given ϵ in the graph induced by the potentials involved in the marginalisation are removed prior to marginalisation.

Once the message passing process is completed, the marginal of each $X \in \mathcal{X}$ is computed from any node in $\mathcal{T}$ involving X. It is not necessary to use the same marginalisation operation for both message and posterior marginals computation.

3. The *BreakTies* algorithm

Madsen [20] reports on experiments using different cost measures in LPAR to select the next arc to reverse in the process of eliminating a variable X. Here, we extend this work with an algorithm for breaking ties in the cost measure.

We introduce *BreakTies* (see Algorithm 1) for reversing the outgoing arcs of a variable X in the process of eliminating X from a set of potentials represented as a DAG G. *BreakTies* takes as input a variable X to be eliminated from a DAG and a sequence of cost measures for determining the next arc to reverse. If there is a tie for the best value in a cost measure s_i, the algorithm proceeds in the sequence of cost measures to score the arcs with a tie for best value in the cost measure s_{i+1}. The algorithm proceeds through the sequence until a unique arc is identified or until all cost measures have been considered and an arc to reverse is selected at random. In the process of eliminating a variable by arc-reversal, it is, as mentioned above, important not to induce directed cycles in the directed graphs. Acyclicity is a non-local property.

Data: X is the variable to eliminate next from a set of potentials represented by the DAG G and $s_1, \ldots, s_n$ is a sequence of score functions.
Result: All outgoing arcs of X in G are reversed.
begin
1 **foreach** $Y \in \mathrm{ch}(X)$ **do**
for $i = 1$ **to** n **do** Compute $s_i(X, Y)$
end
2 Set $\mathcal{A}_0 = \{(X, Y) : Y \in \mathrm{ch}(X)\}$
while $|\mathcal{A}_0| > 0$ **do**
for $i = 1$ **to** n **do**
3 Set $\mathcal{A}_i = \arg_{(X,Y)} \min\{s_i(X, Y) : (X, Y) \in \mathcal{A}_{i-1}\}$
end
4 Select $(X, Y) \in \mathcal{A}_n$ randomly
Reverse (X, Y)
Set $\mathcal{A}_0 = \mathcal{A}_0 \setminus \{(X, Y)\}$
end
end

Algorithm 1: *BreakTies* for reversing all outgoing arcs of X.

The loop in Step 1 computes the cost of reversing the arc (X, Y) for each $Y \in \mathrm{ch}(X)$ and for each cost function. In Step 2, the set $\mathcal{A}_0$ is defined as the arcs directed away from variable X before any arc is reversed. This is the starting point for selecting the next arc to reverse until all outgoing arcs of X are reversed and X in effect is eliminated. In Step 3, the subset $\mathcal{A}_i$ of $\mathcal{A}_{i-1}$ (i.e., the remaining outgoing arcs of X) is the set of arcs with minimum cost in the score function s_i and not producing a directed cycle if reversed. Notice that $\mathcal{A}_{i-1}$ may be a unique arc in which case this arc is also unique in $\mathcal{A}_i$. If $|\mathcal{A}_n| = 1$, then a unique arc has been identified. If $|\mathcal{A}_n| > 1$, then the arcs received the same score for each s_i, for $i = 1, 2, \ldots, n$ and an arc is selected at random in Step 4. In this case, the score functions under consideration were not able to break the ties between elements of $\mathcal{A}_n$.

Example 2. With respect to Fig. 2(i), let us revisit Example 1 using *BreakTies* (Algorithm 1) with cost functions sequence $\mathcal{S} = (s_{cptw}, s_{fiw}, s_{fi}, s_{nop})$. Variable X_1 is to be eliminated and has two children X_3 and X_4.

As *BreakTies* with sequence $\mathcal{S}$ will start with *cptw*, scores $s_{cptw}(X_1, X_3)$ and $s_{cptw}(X_1, X_4)$ are the same as in Example 1. Using *BreakTies*, however, given a tie, we proceed to apply the second score function s_{fiw} in $\mathcal{S}$ to decide which arc to reverse first:

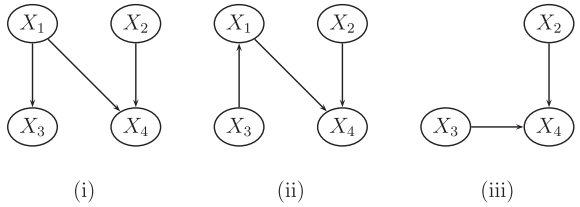

Fig. 2. Eliminating X_1 in (i) by reversing arc (X_1, X_3) (ii) followed by arc (X_1, X_4) (iii).

$$s_{fiw}(X_1, X_3) = w(X_3, X_1) = 8,$$
$$s_{fiw}(X_1, X_4) = w(X_2, X_1) + w(X_4, X_1) = 4 + 4 = 8.$$

To break *fiw* ties, *BreakTies* proceeds to the third cost measure s_{fi}:

$$s_{fi}(X_1, X_3) = 1,$$
$$s_{fi}(X_1, X_4) = 2.$$

As $s_{fi}(X_1, X_3)$ is the lowest, *BreakTies* reverses arc (X_1, X_3) first, as follows:

$$p(X_1, X_3) = p(X_1) \cdot p(X_3|X_1),$$
$$p(X_3) = \sum_{X_1} p(X_1, X_3),$$
$$p(X_1|X_3) = \frac{p(X_1, X_3)}{p(X_3)}.$$

The resulting DAG is shown in Fig. 2(ii). The reversal of (X_1, X_4) yields Fig. 2(iii) by computing

$$p(X_1, X_4|X_2, X_3) = p(X_1|X_3) \cdot p(X_4|X_1, X_2),$$
$$p(X_4|X_2, X_3) = \sum_{X_1} p(X_1, X_4|X_2, X_3).$$

Example 2 illustrates a couple of important points. Note that multiple cost measures were used in the sequence specified by $\mathcal{S}$. First, *cptw* cost measures were computed and found to be tied. Second, *fiw* cost measures were computed and also found to be tied. Third, *fi* cost measures were computed and arc (X_1, X_3) was preferred as it had the lowest score. If *fi* scores had been tied, then *nop* cost measures would have been computed.

The next example shows that randomly breaking ties can lead to unnecessary computation being performed when eliminating sets of variables.

Example 3. Suppose we have to eliminate variable X_2 after variable X_1 has been eliminated in our running example. If we only use a single cost measure, for instance, s_{cptw} and randomly break ties as shown in Example 1, variable X_2 needs to be eliminated from the network in Fig. 3(i). Since

$$s_{cptw}(X_2, X_3) = 2 \cdot 4 \cdot 2 = 16,$$
$$s_{cptw}(X_2, X_4) = 2 \cdot 2 = 4,$$

arc (X_2, X_4) is reversed as (see Fig. 3(i)-(iii))

$$p(X_2, X_4) = p(X_2) \cdot p(X_4|X_2),$$
$$p(X_4) = \sum_{X_2} p(X_2, X_4),$$
$$p(X_2|X_4) = \frac{p(X_2, X_4)}{p(X_4)}.$$

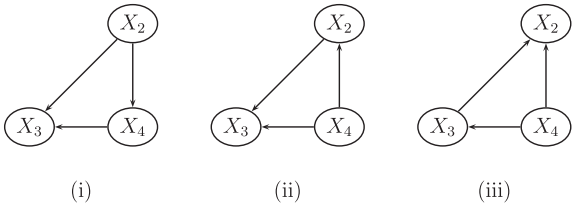

Fig. 3. Eliminating X_2 in (i) by reversing arc (X_2, X_4) (ii) followed by arc (X_2, X_3) (iii).

Arc (X_2, X_3) is then reversed and the variable X_2 is eliminated as

$$p(X_2, X_3|X_4) = p(X_2|X_4) \cdot p(X_3|X_2),$$
$$p(X_3|X_4) = \sum_{X_2} p(X_2, X_3|X_4).$$

On the other hand, if we use *BreakTies* with $\mathcal{S} = (s_{cptw}, s_{fiw}, s_{fi}, s_{nop})$ to eliminate variables using AR in Example 2, variable X_2 needs to be eliminated from the network in Fig. 2 (iii). In this case, variable X_2 has only one child X_4, which allows X_2 to be eliminated more efficiently as

$$p(X_2, X_4|X_3) = p(X_2) \cdot p(X_4|X_2, X_3),$$
$$p(X_4|X_3) = \sum_{X_2} p(X_2, X_4|X_3).$$

Example 3 reveals that randomly breaking s_{cptw} ties can result in extra arcs being added, producing additional unnecessary calculation during belief update. For instance, in our running example, the approach taken by Madsen [20] can add one more arc (X_2, X_3) in Fig. 1(iii) than *BreakTies* with cost functions sequence $\mathcal{S}$ did in Fig. 2(iii). That is, X_2 only has one child using *BreakTies*, whereas X_2 has two children using s_{cptw} alone. Adding children to a variable to be subsequently eliminated means that a single cost measure, e.g., s_{cptw}, in isolation can force more computation to be performed.

4. Experimental results

This section presents the results of an empirical evaluation of the impact of performing arc-reversal operations in different orders when eliminating variables in LPAR. Three different types of evaluations are performed.

4.1. Experimental setup

The experiments were performed on the 25 real-world networks described in Table 1, where $s(C) = \prod_{X \in C} |\text{dom}(X)|$. Junction trees have been generated using the *total weight* heuristic [11]. The test set consists of networks of different size and complexity in terms of the size of the largest clique of the corresponding junction tree.

In the experiments, AR is applied to build messages and VE is applied to compute the posterior marginal of each non-evidence variable. The fill-in-weight heuristic [11] is applied to determine the elimination order when computing messages and posterior marginals. LP computes the posterior marginal distribution of each non-evidence variable in the Bayesian network.

We compute all marginals for randomly generated sets of evidence. We perform experiments with two different approaches to generating evidence. First, we consider evidence set sizes ranging from zero to all variables in the Bayesian network. Ten sets of evidence are randomly generated for each size. Second, we compute the average run time across one hundred sets of randomly generated evidence. The same sets of evidence are used in the experiments when comparing different algorithms.

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 2.6.38-16-server) server with an Intel Xeon (TM) E3-1270 Processor (3.4GHz, 4C/8T and 8MB Cache) and 32 GB RAM.

4.2. New cost measure in isolation

In this section, we report on the performance analysis of using the new cost measure *dw* for selecting the next arc to reverse when eliminating variables using AR.

Table 1
Description of Bayesian networks used in the experiments and their corresponding junction trees.

Id	$\mathcal{N}$	$\lvert\mathcal{X}\rvert$	$\lvert\mathcal{C}\rvert$	$\max_{C\in\mathcal{C}} s(C)$	$\sum_{C\in\mathcal{C}} s(C)$
1	3nt	58	41	2,835	12,848
2	Barley	48	36	7,257,600	17,140,796
3	Diabetes	413	337	84,480	9,825,960
4	Hepar_II	70	58	384	2,617
5	KK	50	38	5,806,080	14,011,466
6	Link	724	576	2,097,152	23,983,922
7	Mildew	35	29	1,249,280	3,400,464
8	Munin1	189	162	38,400,000	83,735,758
9	Munin2	1,003	854	151,200	2,049,942
10	Munin3	1,044	904	156,800	3,077,688
11	Munin4	1,041	877	448,000	8,860,074
12	Water	32	21	589,824	3,028,305
13	andes	223	180	65,536	215,806
14	cc145	145	140	1,024	3,768
15	cc245	245	235	262,144	685,412
16	hailfinder	56	43	3,267	9,406
17	medianus-II	56	44	540,000	1,297,820
18	oow-trad	33	22	2,041,200	6,601,832
19	oow_bas	33	19	510,300	2,006,078
20	oow_solo	40	29	1,644,300	4,651,788
21	pathfinder	109	91	32,256	182,641
22	sacso	2,371	1,229	147,456	960,335
23	ship-ship	50	35	4,032,000	24,258,572
24	system_v57	85	75	69,984	1,373,973
25	win95pts	76	50	512	2,684

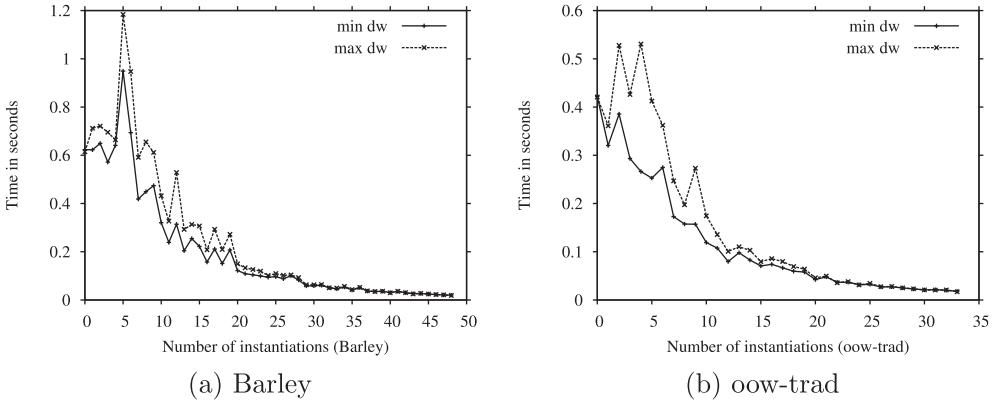

Fig. 4. Time performance in seconds on Barley and oow-trad using *dw*.

Similar to the experiments on using the other cost measures considered for selecting the next arc to reverse in LPAR [20], we compare the performance of LPAR when selecting the next arc to reverse using the minimum *dw* score with the performance when selecting the next arc to reverse using the maximum *dw* score. Fig. 4(a) and (b) shows the average time cost in seconds of belief update in Barley and oow-trad, respectively, as a function of the number of instantiated variables ranging from zero to all variables instantiated. Let us consider a specific example. In Fig. 4(a), for the case of seven evidence variables, the average running time of using min *dw* (best) is 0.4186 s, while the average running time for max *dw* (worst) is 0.5915 s. Hence, using min *dw* in this case produced time savings on average of 29.2% over using the max *dw*.

Table 2 shows the average time cost in seconds of belief update for LARP using min dw and max dw as well as using four other (minimum) cost measures on the 25 real-world networks considered in the experiments. The average is computed over one hundred randomly generated sets of evidence.

It is clear from the experimental results reported here that there can be a significant difference between the performance of LPAR when selecting the next arc to reverse using the minimum dw score and the performance of LPAR when selecting the next arc to reverse using the maximum dw score. Table 2 also shows that most often fiw produces the lowest cost on the networks considered whereas dw (and fi) most often produces the highest cost. In general, the performance of dw is comparable to the performance of the four other cost measures considered in the evaluation. For many networks the difference in performance between the highest and lowest cost is relatively small.

Observe that in ten cases all scores produce the same average time costs. Moreover, in almost all cases, s_{fiw} is among the set of scores producing the lowest average time cost and often outperforms the other scores whereas s_{cptw} is less often

Table 2
Time cost in seconds of belief update for LARP using *dw*, *cptw*, *fi*, *fiw* and *nop*.

$\mathcal{N}$	$\mu_{\min dw}$	$\sigma^2_{\min dw}$	$\mu_{\max dw}$	$\sigma^2_{\max dw}$	μ_{cptw}	σ^2_{cptw}	μ_{fi}	σ^2_{fi}	μ_{fiw}	σ^2_{fiw}	μ_{nop}	σ^2_{nop}
1	0.04	0.00	0.04	0.00	0.04	0.00	0.04	0.00	0.04	0.00	0.04	0.00
2	0.17	0.05	0.21	0.09	0.18	0.07	0.19	0.08	0.16	0.04	0.18	0.07
3	1.51	4.41	2.42	13.68	1.15	2.12	1.32	3.11	1.13	2.02	1.26	2.82
4	0.08	0.00	0.09	0.00	0.08	0.00	0.08	0.00	0.08	0.00	0.08	0.00
5	0.15	0.03	0.18	0.05	0.15	0.02	0.15	0.03	0.14	0.02	0.15	0.02
6	5.27	61.39	13.31	379.18	4.01	33.72	6.31	99.73	4.48	50.27	4.34	44.93
7	0.08	0.01	0.09	0.01	0.08	0.01	0.08	0.01	0.08	0.01	0.08	0.01
8	2.08	48.23	2.81	59.12	1.55	24.96	1.76	31.19	1.56	26.88	1.55	25.51
9	1.48	0.66	2.47	6.38	1.29	0.29	1.31	0.31	1.20	0.17	1.28	0.27
10	3.84	14.85	7.63	75.95	3.40	9.61	3.50	11.13	2.69	5.24	3.34	9.50
11	2.83	6.71	5.73	41.97	2.43	4.30	2.48	4.46	2.08	2.66	2.34	3.50
12	0.09	0.01	0.11	0.01	0.09	0.01	0.09	0.01	0.09	0.01	0.09	0.01
13	0.36	0.03	0.46	0.06	0.36	0.03	0.37	0.04	0.36	0.03	0.36	0.03
14	0.20	0.02	0.23	0.02	0.21	0.02	0.21	0.02	0.21	0.02	0.21	0.02
15	0.47	0.05	0.59	0.10	0.49	0.05	0.50	0.06	0.47	0.05	0.49	0.05
16	0.05	0.00	0.05	0.00	0.05	0.00	0.05	0.00	0.05	0.00	0.05	0.00
17	0.07	0.00	0.08	0.00	0.07	0.00	0.07	0.00	0.07	0.00	0.07	0.00
18	0.11	0.02	0.14	0.03	0.10	0.01	0.10	0.01	0.10	0.01	0.10	0.01
19	0.06	0.00	0.07	0.00	0.06	0.00	0.06	0.00	0.06	0.00	0.06	0.00
20	0.12	0.02	0.15	0.04	0.12	0.02	0.12	0.02	0.12	0.02	0.12	0.02
21	0.22	0.03	0.22	0.03	0.22	0.03	0.22	0.03	0.22	0.03	0.22	0.03
22	4.84	0.49	7.91	53.32	4.79	0.35	4.82	0.39	4.78	0.32	4.78	0.33
23	0.28	0.23	0.36	0.35	0.28	0.20	0.28	0.21	0.26	0.18	0.27	0.20
24	0.13	0.01	0.15	0.01	0.12	0.01	0.12	0.01	0.12	0.01	0.12	0.01
25	0.07	0.00	0.09	0.00	0.07	0.00	0.07	0.00	0.07	0.00	0.07	0.00
Overall	0.98	7.8	1.82	35.72	0.86	4.77	0.97	8.58	0.82	5.22	0.87	5.3

among the scores producing the lowest average score and less often outperforms the other scores. This is contrary to the preliminary findings by [20].

4.3. Best and worst tie breaking and maximum cost

Based on the experimental results reported in Section 4.2, by Madsen [20], and by Butz et al. [3], the following six different sequences $\mathcal{S}_0, \mathcal{S}_1, \ldots, \mathcal{S}_5$ of the five cost measures are now investigated:

- $\mathcal{S}_0 = (s_{cptw}, s_{fiw}, s_{dw}, s_{fi}, s_{nop})$,
- $\mathcal{S}_1 = (s_{dw}, s_{fiw}, s_{cptw}, s_{fi}, s_{nop})$,
- $\mathcal{S}_2 = (s_{fiw}, s_{dw}, s_{cptw}, s_{fi}, s_{nop})$,
- $\mathcal{S}_3 = (s_{fiw}, s_{cptw}, s_{dw}, s_{fi}, s_{nop})$,
- $\mathcal{S}_4 = (s_{cptw}, s_{fiw}, s_{fi}, s_{dw}, s_{nop})$, and
- $\mathcal{S}_5 = (s_{fiw}, s_{nop}, s_{cptw}, s_{dw}, s_{fi})$.

The six sequences of cost measures have been selected based on the nature of the cost measures and results reported previously. We expect a cost measure considering the weight of edges to perform better in terms of breaking ties than a cost measure only counting edges, i.e., we expect *fiw* to perform better than *fi*. The reason is that a cost measure considering the weight of an edge is more fine-grained that a cost measure only counting edges. In fact, we would expect any cost measure considering weights of edges to perform better than any cost measure only counting edges, that is, we would expect *dw* to perform better than *fi*. These considerations and the preliminary experimental analysis reported in [20] suggested the following effectiveness ranking of the four cost measures: *cptw*, *fiw*, *fi* and *nop*. This produced $\mathcal{S}_0$. The sequence $\mathcal{S}_1$ is the *reverse* of $\mathcal{S}_0$, again keeping the above comments in mind. The sequences $\mathcal{S}_2$ and $\mathcal{S}_3$ are similar to $\mathcal{S}_1$ applying *dw* at different starting places in the sequence. Finally, $\mathcal{S}_4$ is $\mathcal{S}_0$ with *dw* and *fi* swapped, whereas $\mathcal{S}_5$ is a sequence created based on the results reported in, i.e., cost measures ordered in reverse according to the number of times they produce the lowest cost.

The aim of the experimental evaluation is to assess the potential impact cost measure sequences can have on the performance of belief update. In addition, the aim of considering more than one sequence is to assess if the potential performance improvement is robust with respect to the cost measure sequence applied.

Three different variants of *BreakTies* are now studied. We compare *BreakTies* using the min-operator in Step 3 ($i = 1, 2, \ldots, n$), *BreakTies* using the max-operator in Step 3 ($i = 1, 2, \ldots, n$) and *BreakTies* using the min-operator for $i = 1$ and the max-operator for $i = 2, 3, \ldots, n$. The first case is referred to as *best tie breaking*, the second case as *maximum cost* and the last case as *worst tie breaking*. In the first and last cases, the algorithm aims to minimise the first score, whereas the subsequent scores are minimised in the first variant and maximised in the last variant. Finally, in the second case, the algorithm aims to maximize the cost.

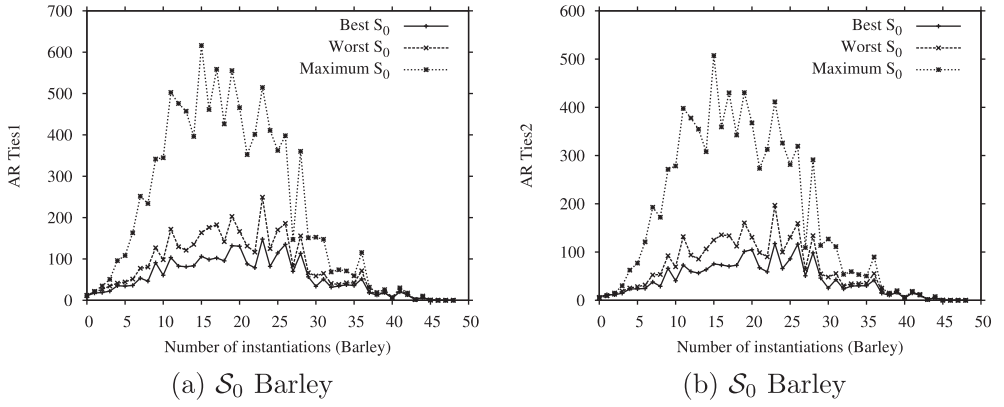

Fig. 5. Number of ties at first and second iterations of Step 3 in *BreakTies* for Barley using $\mathcal{S}_0$.

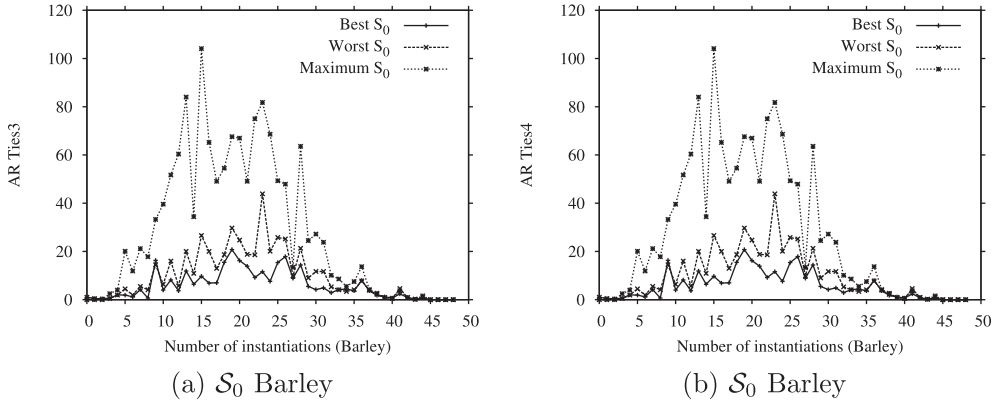

Fig. 6. Number of ties at third and fourth iterations of Step 3 in *BreakTies* for Barley using $\mathcal{S}_0$.

First, a number of examples are included to illustrate the potential of breaking ties using different algorithms. These examples are based on propagating ten different sets of evidence for each size of the evidence set ranging from zero to all variables instantiated. Next, the results of propagating one hundred sets of randomly generated evidence are presented as tables.

Figs. 5(a), 5(b), 6(a) and 6(b) show the number of ties for the first four cost measures of $\mathcal{S}_0$ on Barley as a function of the number of variables instantiated by evidence. The number of ties is equal to the size of $\mathcal{A}_i$ in Step 3 of *BreakTies*. Notice that Fig. 6(a) and (b) show the same values, i.e., no further ties are broken.

The sequence $\mathcal{S}_0 = (s_{cptw}, s_{fiw}, s_{dw}, s_{fi}, s_{nop})$ uses *cptw* as the first score measure. The average number of ties in the first score for Barley is shown in Fig. 5(a), while the average number of ties in the second score (the *fiw* measure) for Barley is shown in Fig. 5(b). Thus, the average number of scores resolved by the second score is the difference between the two scores. For the case of eleven evidence variables in Fig. 5(a), the average number of ties in the first score for *best tie breaking* is 103.4, while the average number of ties for *maximum cost* is 502.6 and the average number of ties for *worst tie breaking* is 171.8. Hence, *best tie breaking* can result in a reduction in the number of ties at the first score of 79.4% and 39.8.% over *maximum cost* and *worst tie breaking*, respectively. The experiments show clearly that a large number of ties are produced for each of the cost measures.

Fig. 7(a) and (b) show the time performance of *BreakTies* with *best tie breaking*, *worst tie breaking* and *maximum cost* using $\mathcal{S}_0$ and $\mathcal{S}_1$ on Barley, respectively. Fig. 7 shows a clear difference in time performance between *best tie breaking* and *maximum cost*. As an example, for the case of five evidence variables in Fig. 7(a), the average running time of using S_0 with *best tie breaking* is 0.8344 s, while the average running time of using S_0 with *maximum cost* is 2.2294 s. Hence, using *best tie breaking* can produce a time savings of 62.6% over using *maximum cost* in *BreakTies*. For five evidence variables the average running time of using S_0 with *worst tie breaking* is 0.9807 s. Hence, using *best tie breaking* can produce a time savings of 14.9% over using *worst tie breaking* in *BreakTies*.

Tables 3–8 show the average run time of belief update for the networks considered in the evaluation across one hundred sets of randomly generated evidence. Table 3 shows the average cost of belief update in the networks using the sequence $\mathcal{S}_0$. For almost all cases, the cost of *best tie breaking* is considerably lower than the cost of *maximum cost*. In addition, the cost of *worst tie breaking* is almost always lower than the cost of *maximum cost* and almost always higher than *best tie breaking*. Table 4 shows the average cost of belief update in the networks using the sequence $\mathcal{S}_1$. The cost of *best tie breaking* is considerably

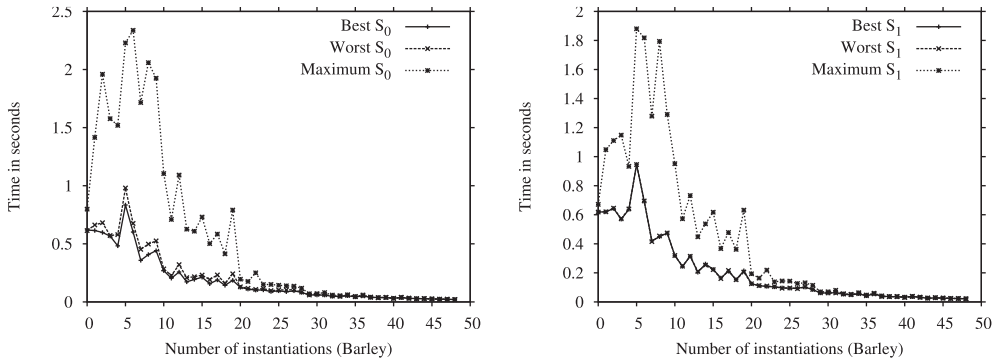

Fig. 7. Time performance in seconds of *BreakTies* using $\mathcal{S}_0$ and $\mathcal{S}_1$ on Barley.

Table 3
Time cost in seconds using $\mathcal{S}_0$ and one hundred randomly generated sets of evidence.

$\mathcal{N}$	μ_{best}	σ^2_{best}	$\mu_{maximum}$	$\sigma^2_{maximum}$	μ_{worst}	σ^2_{worst}
1	0.04	0.00	0.04	0.00	0.04	0.00
2	0.16	0.04	0.52	0.98	0.19	0.08
3	1.14	2.05	3.57	29.69	1.23	2.49
4	0.07	0.00	0.11	0.00	0.08	0.00
5	0.14	0.02	0.34	0.30	0.16	0.03
6	4.08	42.54	43.86	2676.24	4.47	43.28
7	0.08	0.01	0.11	0.01	0.09	0.01
8	1.52	24.81	8.07	622.41	1.59	25.56
9	1.21	0.18	7.44	130.16	1.42	0.46
10	2.70	5.27	18.73	584.72	4.33	17.51
11	1.99	2.22	17.43	610.71	3.01	8.12
12	0.09	0.01	0.16	0.03	0.09	0.01
13	0.35	0.03	0.58	0.11	0.38	0.04
14	0.20	0.02	0.25	0.03	0.22	0.02
15	0.47	0.05	0.79	0.35	0.50	0.06
16	0.05	0.00	0.06	0.00	0.05	0.00
17	0.07	0.00	0.11	0.01	0.08	0.00
18	0.10	0.01	0.19	0.07	0.10	0.01
19	0.06	0.00	0.08	0.00	0.06	0.00
20	0.12	0.02	0.21	0.09	0.12	0.02
21	0.22	0.03	0.23	0.03	0.22	0.03
22	4.76	0.32	11.84	266.56	4.87	0.44
23	0.26	0.18	0.52	0.71	0.29	0.21
24	0.12	0.01	0.17	0.03	0.12	0.01
25	0.07	0.00	0.11	0.00	0.07	0.00

lower than the cost of *maximum cost* for almost all cases, whereas the cost of *worst tie breaking* is always lower than the cost of *maximum cost* and most often higher than *best tie breaking*. Using sequence $\mathcal{S}_2$, Table 5 shows that for almost all cases the cost of *best tie breaking* is considerably lower than the cost of *maximum cost*. Likewise the cost of *worst tie breaking* is always lower than the cost of *maximum cost* and most often higher than *best tie breaking*. Table 6 shows the average cost of belief update in the networks using $\mathcal{S}_3$. For almost all cases, the cost of *best tie breaking* is considerably lower than the cost of *maximum cost*, while the cost of *worst tie breaking* is always lower than the cost of *maximum cost* and most often higher than *best tie breaking*. Using $\mathcal{S}_4$, Table 7 shows that, for almost all cases, the cost of *best tie breaking* is considerably lower than the cost of *maximum cost*. Moreover, the cost of *worst tie breaking* is almost always lower than the cost of *maximum cost* and always higher than *best tie breaking*. Using $\mathcal{S}_5$, Table 8 shows that the cost of *best tie breaking* is considerably lower than the cost of *maximum cost* for almost all cases. Moreover, the cost of *worst tie breaking* is almost always lower than the cost of *maximum cost* and almost always higher than *best tie breaking*.

From the results presented in Tables 3–8, it is clear that in almost all cases the cost of *best tie breaking* is considerably lower than the cost of *maximum cost* whereas the cost of *worst tie breaking* is often lower than the cost of *maximum cost* and almost always higher than *best tie breaking*. Also, in almost all cases the cost of a single measure is higher than the cost of any sequence having the same measure as the first score.

Table 9 shows the average time cost in seconds for *best tie breaking* on the one hundred randomly generated sets of evidence for each network considered in the evaluation. From the table it is clear that S_1 is not once the best sequence and it produces the highest average time cost of performance for almost all examples. None of the other sequences is dominated by or dominates another sequence on these examples.

Table 4
Time cost in seconds using $\mathcal{S}_1$ and one hundred randomly generated sets of evidence.

$\mathcal{N}$	μ_{best}	σ^2_{best}	$\mu_{maximum}$	$\sigma^2_{maximum}$	μ_{worst}	σ^2_{worst}
1	0.04	0.00	0.04	0.00	0.04	0.00
2	0.17	0.05	0.42	0.60	0.17	0.05
3	1.51	4.41	3.39	26.18	1.52	4.56
4	0.08	0.00	0.10	0.00	0.08	0.00
5	0.15	0.03	0.28	0.19	0.15	0.03
6	5.05	55.81	38.03	2004.06	5.44	62.44
7	0.08	0.01	0.10	0.01	0.08	0.01
8	2.07	48.19	5.16	235.58	2.09	49.28
9	1.47	0.64	5.81	72.21	1.48	0.67
10	3.79	14.42	15.51	402.95	3.88	15.34
11	2.81	6.68	14.38	398.24	2.88	7.06
12	0.09	0.01	0.14	0.02	0.09	0.01
13	0.36	0.03	0.57	0.11	0.36	0.03
14	0.21	0.02	0.25	0.03	0.21	0.02
15	0.47	0.05	0.77	0.33	0.48	0.05
16	0.05	0.00	0.06	0.00	0.05	0.00
17	0.07	0.00	0.10	0.01	0.07	0.00
18	0.11	0.02	0.15	0.03	0.11	0.02
19	0.06	0.00	0.07	0.00	0.06	0.00
20	0.12	0.02	0.18	0.06	0.12	0.02
21	0.22	0.03	0.22	0.03	0.22	0.03
22	4.83	0.48	11.23	205.21	4.85	0.52
23	0.28	0.23	0.43	0.46	0.29	0.23
24	0.13	0.01	0.14	0.01	0.13	0.01
25	0.07	0.00	0.11	0.01	0.07	0.00

Table 5
Time cost in seconds using $\mathcal{S}_2$ and one hundred randomly generated sets of evidence.

$\mathcal{N}$	μ_{best}	σ^2_{best}	$\mu_{maximum}$	$\sigma^2_{maximum}$	μ_{worst}	σ^2_{worst}
1	0.04	0.00	0.04	0.00	0.04	0.00
2	0.16	0.04	0.52	0.97	0.16	0.04
3	1.14	2.03	3.57	29.84	1.18	2.16
4	0.08	0.00	0.11	0.00	0.08	0.00
5	0.14	0.02	0.33	0.30	0.15	0.02
6	4.41	48.00	43.89	2727.65	4.54	47.22
7	0.08	0.01	0.11	0.01	0.08	0.01
8	1.55	27.07	7.88	585.42	1.51	24.97
9	1.19	0.15	7.44	130.52	1.25	0.22
10	2.62	4.74	18.72	585.71	2.78	5.26
11	1.99	2.21	17.45	610.30	2.19	2.76
12	0.09	0.01	0.16	0.03	0.09	0.01
13	0.35	0.03	0.57	0.11	0.38	0.04
14	0.20	0.02	0.25	0.03	0.21	0.02
15	0.47	0.05	0.79	0.36	0.49	0.05
16	0.05	0.00	0.06	0.00	0.05	0.00
17	0.07	0.00	0.11	0.01	0.07	0.00
18	0.10	0.01	0.19	0.06	0.10	0.01
19	0.06	0.00	0.08	0.00	0.06	0.00
20	0.12	0.02	0.21	0.09	0.12	0.02
21	0.22	0.03	0.23	0.03	0.22	0.03
22	4.76	0.32	11.83	261.92	4.83	0.39
23	0.26	0.18	0.52	0.68	0.26	0.18
24	0.12	0.01	0.17	0.02	0.12	0.01
25	0.07	0.00	0.11	0.00	0.07	0.00

5. Conclusions

In this paper, we have considered the challenge of ordering AR operations when eliminating variables in LPAR. This work is motivated by the results reported by Madsen [20] on using different cost measures for selecting the next arc to reverse. When using cost measure *cptw* to reverse arcs in LPAR, Madsen [20] will reverse the first arc tied for the lowest score in the event of a tie. Instead, we suggest breaking ties using a sequence of different cost measures.

The *BreakTies* algorithm presented in Section 3 is designed for breaking ties in cost measures when selecting the next arc to reverse in LPAR. It takes as input a sequence of cost measures for reversing an arc and considers in turn a sequence of different measures when all previous measures have produced a tie. The results of the empirical experiments reported in Section 4 illustrate the potential and importance of breaking ties in AR operations. The results show that in almost all cases the cost of a single measure is higher than the cost of any sequence having the same measure as the first score. That is to

Table 6
Time cost in seconds using $\mathcal{S}_3$ and one hundred randomly generated sets of evidence.

$\mathcal{N}$	μ_{best}	σ^2_{best}	$\mu_{maximum}$	$\sigma^2_{maximum}$	μ_{worst}	σ^2_{worst}
1	0.04	0.00	0.04	0.00	0.04	0.00
2	0.16	0.04	0.52	0.97	0.16	0.04
3	1.13	2.00	3.57	29.78	1.18	2.18
4	0.07	0.00	0.11	0.00	0.08	0.00
5	0.14	0.02	0.33	0.30	0.15	0.02
6	4.36	46.21	43.89	2719.13	4.59	48.05
7	0.08	0.01	0.11	0.01	0.08	0.01
8	1.56	27.17	7.85	579.09	1.50	24.68
9	1.19	0.15	7.43	129.91	1.25	0.22
10	2.61	4.68	18.72	584.70	2.79	5.27
11	1.99	2.22	17.47	612.45	2.20	2.81
12	0.09	0.01	0.16	0.03	0.09	0.01
13	0.35	0.03	0.57	0.11	0.38	0.04
14	0.20	0.02	0.25	0.03	0.22	0.02
15	0.47	0.05	0.79	0.35	0.49	0.05
16	0.05	0.00	0.06	0.00	0.05	0.00
17	0.07	0.00	0.11	0.01	0.07	0.00
18	0.10	0.01	0.19	0.07	0.10	0.01
19	0.06	0.00	0.08	0.00	0.06	0.00
20	0.12	0.02	0.21	0.09	0.12	0.02
21	0.22	0.03	0.23	0.03	0.22	0.03
22	4.76	0.31	11.82	263.75	4.84	0.39
23	0.26	0.18	0.52	0.68	0.26	0.18
24	0.12	0.01	0.17	0.03	0.12	0.01
25	0.07	0.00	0.11	0.01	0.07	0.00

Table 7
Time cost in seconds using $\mathcal{S}_4$ and one hundred randomly generated sets of evidence.

$\mathcal{N}$	μ_{best}	σ^2_{best}	$\mu_{maximum}$	$\sigma^2_{maximum}$	μ_{worst}	σ^2_{worst}
1	0.04	0.00	0.04	0.00	0.04	0.00
2	0.16	0.04	0.52	0.98	0.19	0.08
3	1.14	2.06	3.57	29.76	1.23	2.48
4	0.07	0.00	0.11	0.00	0.08	0.00
5	0.14	0.02	0.34	0.30	0.16	0.03
6	3.88	36.07	43.67	2653.36	4.61	46.78
7	0.08	0.01	0.11	0.01	0.09	0.01
8	1.52	24.80	8.07	623.45	1.59	25.54
9	1.21	0.18	7.35	126.51	1.42	0.46
10	2.70	5.25	18.70	583.63	4.34	17.62
11	1.99	2.24	17.42	607.43	3.01	8.16
12	0.09	0.01	0.16	0.03	0.09	0.01
13	0.36	0.03	0.57	0.11	0.38	0.04
14	0.20	0.02	0.25	0.03	0.22	0.02
15	0.47	0.05	0.78	0.35	0.50	0.06
16	0.05	0.00	0.06	0.00	0.05	0.00
17	0.07	0.00	0.11	0.01	0.07	0.00
18	0.10	0.01	0.19	0.07	0.10	0.01
19	0.06	0.00	0.08	0.00	0.06	0.00
20	0.12	0.02	0.21	0.09	0.12	0.02
21	0.22	0.03	0.23	0.03	0.22	0.03
22	4.76	0.33	11.79	258.52	4.87	0.45
23	0.26	0.18	0.52	0.71	0.29	0.21
24	0.12	0.01	0.17	0.03	0.12	0.01
25	0.07	0.00	0.11	0.00	0.07	0.00

say, *BreakTies* improves performance. Section 4 shows also that the new *dw* score is a relevant score for selecting the next arc to reverse in a variable elimination operation by arc-reversal. There is a significant difference between using the min *dw* and the max *dw* score to select the next arc to reverse. The performance of *dw* is compared to the performance of the other four cost measures considered. Its performance is neither better nor worse than the other cost measures except for min *fiw* which most often has the best performance. The performance of LPAR is relatively robust to the selection of cost measure with *fiw* showing the best performance on the networks and sets of random evidence considered in this evaluation.

The results show that there is a significant difference between using the min-operator (*best tie breaking*) and the max-operator (*maximum cost*) in Step 3 of *BreakTies*. A number of examples illustrate that there can be a large difference in the number of ties produced in the process of eliminating variables using different AR orders. Furthermore, the results also show that the performance of *best tie breaking* is almost always better than the performance of *worst tie breaking*. This means that careful tie breaking when reversing arcs in LPAR can improve time performance.

Table 8
Time cost in seconds using $\mathcal{S}_5$ and one hundred randomly generated sets of evidence.

$\mathcal{N}$	μ_{best}	σ^2_{best}	$\mu_{maximum}$	$\sigma^2_{maximum}$	μ_{worst}	σ^2_{worst}
1	0.04	0.00	0.04	0.00	0.04	0.00
2	0.16	0.04	0.52	0.97	0.16	0.04
3	1.14	2.02	3.58	29.88	1.18	2.16
4	0.07	0.00	0.11	0.00	0.08	0.00
5	0.14	0.02	0.34	0.30	0.15	0.02
6	4.11	40.77	43.80	2708.86	4.68	50.10
7	0.08	0.01	0.11	0.01	0.08	0.01
8	1.56	27.20	7.88	585.07	1.50	24.97
9	1.19	0.16	7.43	129.67	1.25	0.22
10	2.62	4.72	18.74	586.13	2.79	5.26
11	1.99	2.22	17.45	610.82	2.19	2.78
12	0.09	0.01	0.16	0.03	0.09	0.01
13	0.36	0.03	0.57	0.11	0.38	0.04
14	0.20	0.02	0.25	0.03	0.22	0.02
15	0.47	0.05	0.79	0.36	0.50	0.05
16	0.05	0.00	0.06	0.00	0.05	0.00
17	0.07	0.00	0.11	0.01	0.07	0.00
18	0.10	0.01	0.19	0.06	0.10	0.01
19	0.06	0.00	0.08	0.00	0.06	0.00
20	0.12	0.02	0.21	0.09	0.12	0.02
21	0.22	0.03	0.22	0.03	0.22	0.03
22	4.76	0.31	11.82	261.88	4.84	0.39
23	0.26	0.18	0.52	0.68	0.26	0.18
24	0.12	0.01	0.17	0.03	0.12	0.01
25	0.07	0.00	0.11	0.00	0.07	0.00

Table 9
Time cost in seconds for *best tie breaking* on one hundred randomly generated sets of evidence.

$\mathcal{N}$	$\mu_{\mathcal{S}_0}$	$\mu_{\mathcal{S}_1}$	$\mu_{\mathcal{S}_2}$	$\mu_{\mathcal{S}_3}$	$\mu_{\mathcal{S}_4}$	$\mu_{\mathcal{S}_5}$
1	0.04	0.04	0.04	0.04	0.04	0.04
2	0.16	0.17	0.16	0.16	0.16	0.16
3	1.14	1.51	1.14	1.13	1.14	1.14
4	0.07	0.08	0.08	0.07	0.07	0.07
5	0.14	0.15	0.14	0.14	0.14	0.14
6	4.08	5.05	4.41	4.36	3.88	4.11
7	0.08	0.08	0.08	0.08	0.08	0.08
8	1.52	2.07	1.55	1.56	1.52	1.56
9	1.21	1.47	1.19	1.19	1.21	1.19
10	2.70	3.79	2.62	2.61	2.70	2.62
11	1.99	2.81	1.99	1.99	1.99	1.99
12	0.09	0.09	0.09	0.09	0.09	0.09
13	0.35	0.36	0.35	0.35	0.36	0.36
14	0.20	0.21	0.20	0.20	0.20	0.20
15	0.47	0.47	0.47	0.47	0.47	0.47
16	0.05	0.05	0.05	0.05	0.05	0.05
17	0.07	0.07	0.07	0.07	0.07	0.07
18	0.10	0.11	0.10	0.10	0.10	0.10
19	0.06	0.06	0.06	0.06	0.06	0.06
20	0.12	0.12	0.12	0.12	0.12	0.12
21	0.22	0.22	0.22	0.22	0.22	0.22
22	4.76	4.83	4.76	4.76	4.76	4.76
23	0.26	0.28	0.26	0.26	0.26	0.26
24	0.12	0.13	0.12	0.12	0.12	0.12
25	0.07	0.07	0.07	0.07	0.07	0.07
Overall	0.8	0.97	0.81	0.81	0.79	0.8

Table 9 shows the cost of *best tie breaking* for the six different sequences considered in the evaluation. The results in the table show that on the randomly generated evidence sets considered, the sequence $\mathcal{S}_1$ most often has a higher cost than the other measures. Except for this case, the empirical analysis reported here does not indicate that one AR tie-breaking sequence is superior or inferior to another. That is, the performance improvement is robust with respect to the cost measure sequence applied. The results given in this paper demonstrate that using *BreakTies* to break ties when determining the next arc to reverse in a variable elimination operation using AR can have a positive impact on time performance.

Acknowledgments

This research is supported in part by NSERC Discovery Grant 238880. We would like to thank the reviewers for insightful and highly valuable comments that have greatly improved the quality of this paper.

References

[1] C. Butz, J. Chen, K. Konkel, P. Lingras, A formal comparison of variable elimination and arc reversal in bayesian network inference, Intelligent Decision Technologies 3 (2009) 173–180.
[2] C. Butz, K. Konkel, P. Lingras, Join tree propagation utilizing both arc reversal and variable elimination, International Journal of Approximate Reasoning 52 (2011) 948–959.
[3] C.J. Butz, A.L. Madsen, K. Williams, Using four cost measures to determine arc reversal orderings, in: Proceedings of the European Conferences on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 2011, pp. 110–121.
[4] E. Cinicioglu, P. Shenoy, Arc reversals in hybrid bayesian networks with deterministic variables, International Journal of Approximate Reasoning 52 (2011) 948–959.
[5] G.F. Cooper, The computational complexity of probabilistic inference using Bayesian belief networks, Artificial Intelligence 42 (1990) 393–405.
[6] R.G. Cowell, A.P. Dawid, S.L. Lauritzen, D.J. Spiegelhalter, Probabilistic Networks and Expert Systems, Springer-Verlag, 1999.
[7] A. Darwiche, Modeling and Reasoning with Bayesian Networks, Cambridge University Press, 2009.
[8] R. Dechter, Bucket elimination: a unifying framework for probabilistic inference, in: Proceedings of the Conference on Uncertainty in Artificial Intelligence, 1996, pp. 211–219.
[9] E. Ejsing, P. Vastrup, A.L. Madsen, Probability of default for large corporates, Bayesian Networks: A Practical Guide to Applications, Wiley, New York, 2008, pp. 329–344.
[10] D. Geiger, T. Verma, J. Pearl, Identifying independence in Bayesian networks, Networks 20 (1990) 507–534.
[11] F. Jensen, HUGIN API Reference Manual, www.hugin.com. Version 7.6, 2012.
[12] F.V. Jensen, S.L. Lauritzen, K.G. Olesen, Bayesian updating in causal probabilistic networks by local computations, Computational Statistics Quarterly 4 (1990) 269–282.
[13] F.V. Jensen, T.D. Nielsen, Bayesian Networks and Decision Graphs, second ed., Springer, 2007.
[14] U.B. Kjærulff, A.L. Madsen, Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis, second ed., Springer, 2013.
[15] D. Koller, N. Friedman, Probabilistic Graphical Models – Principles and Techniques, MIT Press, 2009.
[16] S.L. Lauritzen, D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, Journal of the Royal Statistical Society, Series B 50 (1988) 157–224.
[17] Z. Li, B. D'Ambrosio, Efficient inference in bayes networks as a combinatorial optimization problem, International Journal of Approximate Reasoning 11 (1994) 55–81.
[18] A.L. Madsen, An empirical evaluation of possible variations of lazy propagation, in: Proceedings of the Conference on Uncertainty in Artificial Intelligence, 2004, pp. 366–373.
[19] A.L. Madsen, Variations over the message computation algorithm of lazy propagation, IEEE Transactions on Systems, Man, and Cybernetics Part B 36 (2006) 636–648.
[20] A.L. Madsen, Improvements to message computation in lazy propagation, International Journal of Approximate Reasoning 51 (2010) 499–514.
[21] A.L. Madsen, F.V. Jensen, Lazy propagation: a junction tree inference algorithm based on lazy evaluation, Artificial Intelligence 113 (1999) 203–245.
[22] R. Neapolitan, Probabilistic Methods for Bioinformatics with an Introduction to Bayesian Networks, Morgan Kaufman, New York, 2009.
[23] S. Olmsted, On representing and solving decision problems, Ph.D. Thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA, 1983.
[24] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Series in Representation and Reasoning, Morgan Kaufman Publishers, San Mateo, CA, 1988..
[25] O. Pourret, P. Naim, B.E. Marcot, Bayesian Networks: A Practical Guide to Applications, Wiley, New York, 2008.
[26] R. Shachter, B. D'Ambrosio, B. DelFavero, Symbolic probabilistic inference in belief networks, in: Proceedings of the Eighth National Conference on AI, 1990, pp. 126–131.
[27] R.D. Shachter, Evaluating influence diagrams, Operations Research 34 (1986) 871–882.
[28] G. Shafer, Probabilistic Expert Systems, SIAM, 1996.
[29] P.P. Shenoy, Binary join trees for computing marginals in the Shenoy–Shafer architecture, International Journal of Approximate Reasoning 17 (1997) 239–263.
[30] P.P. Shenoy, G. Shafer, Axioms for probability and belief-function propagation, in: Proceedings of the Conference on Uncertainty in Artificial Intelligence, 1990, pp. 169–198.
[31] W.X. Wen, Optimal decomposition of belief networks, in: Proceedings of the Conference on Uncertainty in Artificial Intelligence, 1991, pp. 209–224.
[32] M. Yannakakis, Computing the minimum fill-in is NP-complete, SIAM Journal of Algebraic and Discrete Methods 2 (1981) 77–79.
[33] N.L. Zhang, D. Poole, A simple approach to Bayesian network computations, in: Proceedings of the Canadian Conference on Artificial Intelligence, 1994, pp. 171–178.