Using Four Cost Measures to Determine Arc Reversal Orderings

Cory J. Butz[1], Anders L. Madsen[2], and Kevin Williams[1]

[1] Department of Computer Science, University of Regina,
Regina, Saskatchewan, S4S 0A2 Canada
{butz,willikev}@cs.uregina.ca
[2] HUGIN EXPERT A/S, Gasværksvej 5, DK-9000 Aalborg, Denmark
Anders.L.Madsen@hugin.com

Abstract. Four cost measures s_1, s_2, s_3, s_4 were recently studied for sorting the operations in Lazy propagation with arc reversal (LPAR), a join tree propagation approach to Bayesian network inference. It has been suggested to use s_1 with LPAR, since there is an effectiveness ranking, say s_1, s_2, s_3, s_4, when applied in isolation. In this paper, we also suggest to use s_1 with LPAR, but to use s_2 to break s_1 ties, s_3 to break s_2 ties, and s_4 to break s_3 ties. Experimental results show that sometimes there is a noticeable gain to be made.

Keywords: Bayesian networks, arc reversal, join tree propagation.

1 Introduction

Bayesian networks [6,8,9,10,18] provide a rigorous foundation for uncertainty management by combining probability theory and graph theory. They have been applied in practice to a number of problem domains [19], including bioinformatics [16]. 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 Cooper [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) [17,20] removes 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) [1], eliminates a variable by multiplying together all of the distributions involving the variable and then summing the variable out of the obtained product. *Join tree propagation*, which Shafer [21] emphasizes is central to the theory and practice of probabilistic expert systems, first builds a secondary network, called a join tree, from the DAG of the Bayesian network and then performs inference by propagating probabilities in the join tree [2,3,4]. Madsen and

W. Liu (Ed.): ECSQARU 2011, LNAI 6717, pp. 110–121, 2011.

Jensen [15] significantly advanced the field of join tree propagation with *Lazy Propagation* (LP), which maintains a factorization of distributions allowing for *barren* variables [20] and independencies induced by evidence to be exploited. Madsen [12,13] modified LP by replacing VE as the message construction algorithm with AR, giving LPAR, and conducted an empirical study of LP and LPAR. Very recently, Madsen [14] demonstrated that the order in which arcs are reversed in LPAR can affect the amount of computation needed. Experimental results suggest that *cpt-weight* (*cptw*) is the best of four measures [14].

Here we advocate using all four cost measures in ranked order starting with *cptw*. Our analysis in Section 4 shows that *cptw* seems to be the closest to optimizing the number of arithmetic operations as it is focused on reducing the size of the CPTs constructed. This is similar to reducing the weight of cliques when building join trees [10]. In the event of a *cptw* tie, reverse the best arc as determined by the second-best cost measure, *fill-in weight* (*fiw*). Similarly, proceed to the third-best cost measure *fill-in* (*fi*) to break *fiw* ties, and to *number-of-parents* (*nop*) to break *fi* ties. We illustrate in Section 4 how our approach can save computation, since [14] will reverse the first arc tied for the best *cptw* score. Our experiments in Section 5 show a small but observable computational improvement using five real-world BNs and three randomly-generated BNs.

The remainder is organized as follows. Section 2 contains definitions. We present our new approach in Section 3. Section 4 provides analysis. Experimental findings are given in Section 5. Conclusions are drawn in Section 6.

2 Definitions

Results from Bayesian networks, AR, and four AR cost measures, are reviewed.

2.1 Bayesian Networks

Consider a finite set of discrete random variables $U = \{v_1, v_2, \ldots, v_n\}$. Let $dom(v_i)$ denote the finite domain of values that each variable $v_i \in U$ can assume. For a subset $X \subseteq U$, the Cartesian product of the domains of the individual variables in X is $dom(X)$. An element $x \in dom(X)$ is a *configuration* or *row* of X. A *potential* [21] on $dom(X)$ is a function ϕ such that $\phi(x) \geq 0$, for each configuration $x \in dom(X)$, and at least one $\phi(x)$ is positive. For simplicity we speak of a potential as defined on X instead of on $dom(X)$, and we call X its domain rather than $dom(X)$ [21]. A *joint probability distribution* [21] on U, written $p(U)$, is a function p on U satisfying the following two conditions: (i) $0 \leq p(u) \leq 1$, for each configuration $u \in dom(U)$; (ii) $\sum_{u \in dom(U)} p(u) = 1$. Let X and Y be two disjoint subsets of U. A *conditional probability table* (CPT) [21] for Y given X, denoted $p(Y|X)$, is a nonnegative function on $X \cup Y$, satisfying the following condition: for each configuration $x \in dom(X)$, $\sum_{y \in dom(Y)} p(Y = y \mid X = x) = 1$.

A discrete *Bayesian network* [18] on $U = \{v_1, v_2, \ldots, v_n\}$ is a pair (D, C). D is a DAG with vertex set U. C is the set of CPTs $\{p(v_i|P_i) \mid i = 1, 2, \ldots, n\}$, where P_i denotes the parents of variable $v_i \in D$. For example, one Bayesian

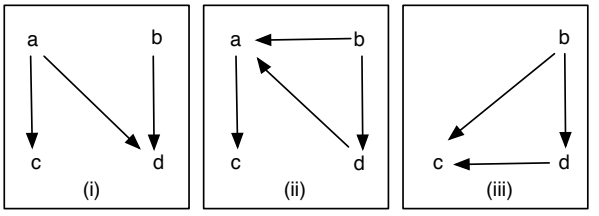

Fig. 1. Eliminating a in (i) by reversing arc (a, d) (ii) followed by arc (a, c) (iii)

network is the DAG in Figure 1 (i) together with CPTs $p(a)$, $p(b)$, $p(c|a)$ and $p(d|a, b)$. Here, the parents P_i of variable $v_i = d$ are $P_i = \{a, b\}$.

The *family* F_i of a variable v_i in a Bayesian network is the variable together with its parents, that is, $\{v_i\} \cup P_i$. By $p(X|Y)$, we always mean $p(X|Y - X)$. We use WZ to mean $W \cup Z$.

2.2 Arc Reversal

Arc reversal (AR) [17,20] eliminates a variable v_i by reversing the arcs (v_i, v_j) for each child v_j of v_i, where $j = 1, 2, \ldots, k$. With respect to multiplication, addition, and division, AR reverses one arc (v_i, v_j) as a three-step process:

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

$$p(v_j|P_iP_j) = \sum_{v_i} p(v_i, v_j|P_iP_j), \tag{2}$$

$$p(v_i|P_iP_jv_j) = \frac{p(v_i, v_j|P_iP_j)}{p(v_j|P_iP_j)}. \tag{3}$$

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

2.3 Four Cost Measures for AR Child Orderings

The following is taken from Madsen [14]. The elimination of variable v_i by a sequence of AR operations produces an ordering of the v_i's children. 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 p' leading to the best time and space performance should be chosen. Since it is not possible by local computations only to identify

the sequence p' having the best time and space cost, the focus is on identifying the sequence ρ with minimum local cost.

Four different score functions for computing the cost of a child ordering ρ are considered: *cptw* (s_1), *fiw* (s_2), *fi* (s_3), and *nop* (s_4).

The cost of reversing arc (v_i, v_j) using the *cptw* score s_1 is:

$$s_1(v_i, v_j) = \prod_{v_k \in F_i F_j} |dom(v_k)|.$$

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

Using the *fiw* score s_2, the cost of reversing (v_i, v_j) is:

$$s_2(v_i, v_j) = \sum_{v_k \in F_j - F_i} w(v_k, v_i) + \sum_{v_l \in P_i - P_j} w(v_l, v_j),$$

where $w(v_a, v_b) = |dom(v_a)| \cdot |dom(v_b)|$. Then, *fiw* cost is equal to the sum of the edge weights of the new parents $F_j - F_i$ of v_i and $P_i - P_j$ of v_j.

The cost of reversing arc (v_i, v_j) using the *fi* score s_3 is:

$$s_3(v_i, v_j) = |F_j - F_i| + |P_i - P_j|,$$

namely, the *fi* cost is equal to the number of edges induced by the new parents $F_j - F_i$ of v_i and the new parents $P_i - P_j$ of v_j.

Using the *nop* score s_4, the cost of reversing (v_i, v_j) is:

$$s_4(v_i, v_j) = |P_i \cup F_j|,$$

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

Experimental results suggest an effectiveness ranking of s_1, s_2, s_3, s_4 [14].

Example 1. Consider eliminating variable a in Fig. 1 (i), where a, b, d are binary and c's domain has four values. Compute the *cptw* score of arcs (a, c) and (a, d) corresponding to the children c and d of a:

$$s_1(a, c) = \prod_{v_k \in \{a,c\}} |dom(v_k)| \;=\; 2 \cdot 4 \;=\; 8,$$
$$s_1(a, d) = \prod_{v_k \in \{a,b,d\}} |dom(v_k)| \;=\; 2 \cdot 2 \cdot 2 \;=\; 8.$$

Since $s_1(a, c)$ is equal to $s_1(a, d)$, *cptw* does not distinguish between arcs (a, c) and (a, d). Thus, one arc is randomly chosen, say (a, d), and reversed:

$$p(a, d|b) = p(a) \cdot p(d|a, b),$$
$$p(d|b) = \sum_a p(a, d|b),$$
$$p(a|b, d) = \frac{p(a, d|b)}{p(d|b)}.$$

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

$$p(a, c|b, d) = p(a|b, d) \cdot p(c|a),$$
$$p(c|b, d) = \sum_{a} p(a, c|b, d).$$

3 AR Child Orderings Using Four Cost Measures

While Madsen's [14] experimental results suggest that *cptw* may the best choice of the four cost measures, our extension, called *BreakTies* and given below, is to use *cptw* as our first cost measure but to rely on other cost measures to break ties. In other words, follow the fixed order s_1, s_2, s_3, s_4 to select the next arc to reverse, only progressing to the next cost measure to break ties.

Algorithm 1. BreakTies (v_i, D)
Input: v_i is the variable to be eliminated in DAG D
Output: the arc (v_i, v_j) in D to be reversed next.
begin
for each remaining child v_j of v_i
  compute $s_1(v_i, v_j)$
  if unique lowest $s_1(v_i, v_j)$
    reverse arc (v_i, v_j)
  else
    for each v_j tying for lowest $s_1(v_i, v_j)$
      compute $s_2(v_i, v_j)$
      if unique lowest $s_2(v_i, v_j)$
        reverse arc (v_i, v_j)
      else
        for each v_j tying for lowest $s_2(v_i, v_j)$
          compute $s_3(v_i, v_j)$
          if unique lowest $s_3(v_i, v_j)$
            reverse arc (v_i, v_j)
          else
            for each v_j tying for lowest $s_3(v_i, v_j)$
              compute $s_4(v_i, v_j)$
              if unique lowest $s_4(v_i, v_j)$
                reverse arc (v_i, v_j)
              else
                randomly pick an arc (v_i, v_j)
return (v_i, v_j)
end

Example 2. With respect to Fig. 2 (i), let us revisit Example 1 using BreakTies. Variable a is to be eliminated and has two children c and d. As BreakTies always

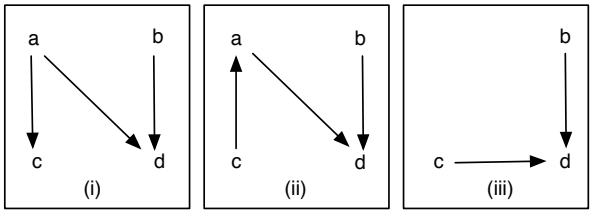

Fig. 2. Eliminating a in (i) by reversing arc (a, c) (ii) followed by arc (a, d) (iii)

starts with *cptw*, scores $s_1(a, c)$ and $s_1(a, d)$ are the same as before. In BreakTies, however, given a tie, we proceed to the second *fiw* score s_2 to decide which arc to reverse first:

$$\begin{aligned} s_2(a, c) &= w(c, a) = 8, \\ s_2(a, d) &= w(b, a) + w(d, a) = 4 + 4 = 8. \end{aligned}$$

To break *fiw* ties, BreakTies proceeds to the *fi* cost measure s_3:

$$\begin{aligned} s_3(a, c) &= 1, \\ s_3(a, d) &= 2. \end{aligned}$$

As $s_3(a, c)$ is the lowest, BreakTies reverses arc (a, c) first, as follows:

$$\begin{aligned} p(a, c) &= p(a) \cdot p(c|a), \\ p(c) &= \sum_a p(a, c), \\ p(a|c) &= \frac{p(a, c)}{p(c)}. \end{aligned}$$

The resulting DAG is shown in Figure 2 (ii). The reversal of (a, d) yields Figure 2 (iii) by computing

$$\begin{aligned} p(a, d|b, c) &= p(a|c) \cdot p(d|a, b), \\ p(d|b, c) &= \sum_a p(a, d|c, d). \end{aligned}$$

Example 2 illustrates a couple of important points. Note that multiple cost measures were used. 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 (a, c) was preferred as it had the lowest score. If *fi* scores had been tied, then *nop* cost measures would have been computed. Also notice how BreakTies starts with *cptw*, considered to be the best, and ends with *nop*, considered to be the weakest measure.

4 Analysis

In this section, we first analyze the *cptw* score with respect to one arc reversal. Next, we scrutinize *cptw* with respect to the overall process of Bayesian network inference using AR.

Bayesian network inference involves the elimination of a set X of variables. To eliminate each variable v_i in X, arc reversal needs to reverse the arc between v_i and each of its children v_j. Consider the amount of computation needed to reverse one arc (v_i, v_j). It can be seen that: (i) the number of multiplications in the first step is equal to $|dom(F_iF_j)|$; (ii) the number of additions in the second step is equal to $|dom(v_i) - 1| \cdot |dom(F_iF_j - v_i)|$; and, (iii) the number of divisions in the third step is equal to $|dom(F_iF_j)|$. This means that the amount of computation needed to reverse one arc is tied directly to *cptw*.

Now let us turn our attention away from reversing one arc to the larger problem of eliminating a set X of variables. The next example shows that randomly breaking *cptw* ties can cost more computation in the long run.

Example 3. Suppose we have to eliminate variable b after variable a has been eliminated in our running example.

If we only use *cptw* and randomly break ties as shown in Example 1, variable b needs to be eliminated from Figure 1 (iii). Since

$$\begin{aligned} s_1(b,c) &= 2 \cdot 4 \cdot 2 \;=\; 16, \\ s_1(b,d) &= 2 \cdot 2 \;=\; 4, \end{aligned}$$

arc (b, d) is reversed as

$$\begin{aligned} p(b,d) &= p(b) \cdot p(d|b), \\ p(d) &= \sum_b p(b,d), \\ p(d|b) &= \frac{p(b,d)}{p(d)}. \end{aligned}$$

Arc (b, c) is then reversed as

$$\begin{aligned} p(b,c|d) &= p(b|d) \cdot p(c|b), \\ p(c|d) &= \sum_b p(b,c|d). \end{aligned}$$

On the other hand, if we use multiple cost measures as done in Example 2, variable b needs to be eliminated from Figure 2 (iii). In this case, variable b only has one child d, which allows b to be eliminated more economically as

$$\begin{aligned} p(b,d|c) &= p(b) \cdot p(d|b,c), \\ p(d|c) &= \sum_b p(b,d|c). \end{aligned}$$

Example 3 reveals that randomly breaking *cptw* ties can result in extra arcs being added. For instance, in our running example, the approach in [14] can add one more arc (b, c) in Fig. 1 (iii) than BreakTies did in Fig. 2 (iii). That is, b only has one child using BreakTies, whereas b has two children using *cptw* alone. Adding children to a variable to be subsequently removed means that the *cptw* cost measure in isolation can force more computation to be performed.

5 Experimental Results

The experiments use the LPAR method in [14], namely, AR is applied to build all messages and VE is applied to compute posterior marginals. The measure *fiw* is used to determine the elimination order and to compute posteriors. Experiments were conducted on 15 real-world networks and 30 randomly generated networks, but we report only on five real-world networks, called Barley [11], KK [11][1], Mildew[2], OOW [7], and ship-ship [7], and three randomly generated networks, called net100, net125, and net150 [14], which are all described in Table 1. For each size of evidence set, ten sets of evidence are generated, with the same evidence used in different runs. To reflect the potential time savings of breaking ties, the *best* and *worst* arcs are reversed based on the next cost measure in BreakTies. Figs. 3 - 6 show running times in seconds on our eight Bayesian networks.

Table 1. Description of test Bayesian networks and corresponding join tree nodes $\mathcal{N}$

BN	$\lvert U\rvert$	$\lvert\mathcal{N}\rvert$	max $\lvert dom(\mathcal{N})\rvert$	total size
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	40	29	1,644,300	4,651,788
ship-ship	50	35	4,032,000	24,258,572
net100	100	85	98,304	311,593
net125	125	109	165,888	408,889
net150	150	131	3,538,944	9,946,960

Let us look at a couple of cases. In Fig. 5 (left), for the case of six evidence variables, the average running time of breaking ties (best) is 1.6866 seconds, while the average running time for breaking ties (worst) is 2.0792 seconds. Hence, breaking ties (best) can result in a time savings of 18.9%. Now consider twenty evidence variables in Fig. 6 (right). The average running time of breaking ties (worst) is 1.5247, which can be bettered by 29.61% to 1.0739 when breaking ties (best). When considering thirty evidence variables in Fig. 6 (right), the results are even more promising. Here a time savings of 42.5% can be obtained when the running time is decreased from 1.9465 (worst) down to 1.1196 (best). While this demonstrates that there can be on average fewer multiplication and division

[1] KK is a preliminary version of Barley.
[2] A network developed by Finn V. Jensen, Jørgen Olesen and Uffe Kjærulff.

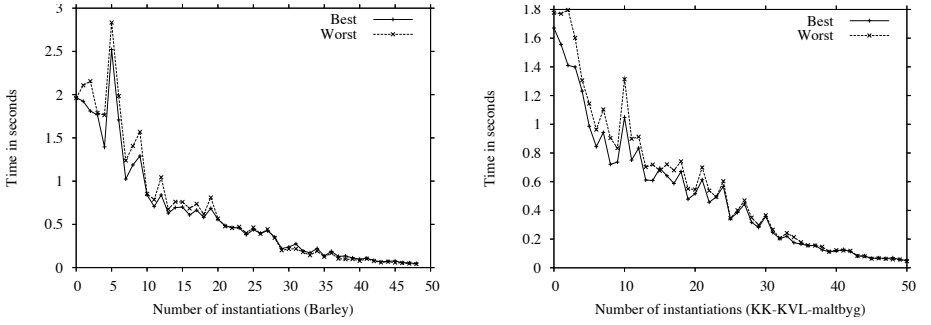

Fig. 3. Time savings of breaking ties by reversing the best and worst arcs as determined by the next cost measure in BreakTies on Barley (left) and KK (right)

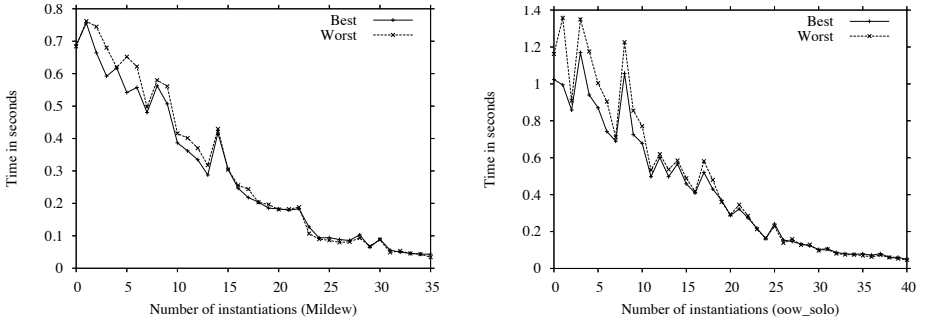

Fig. 4. Time savings of breaking ties by reversing the best and worst arcs as determined by the next cost measure in BreakTies on Mildew (left) and OOW (right)

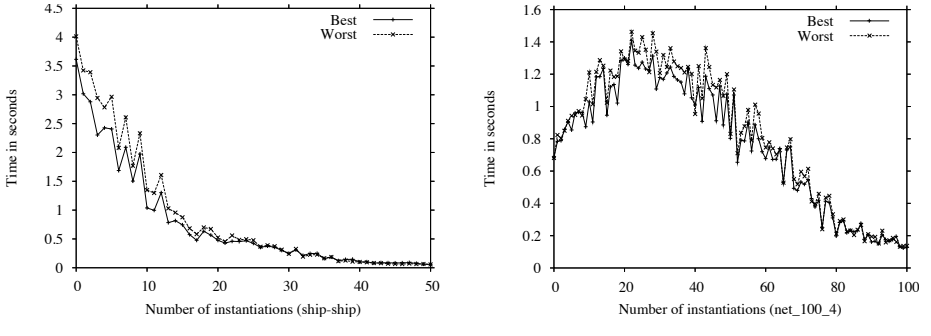

Fig. 5. Time savings of breaking ties by reversing the best and worst arcs as determined by the next cost measure in BreakTies on ship-ship (left) and net100 (right)

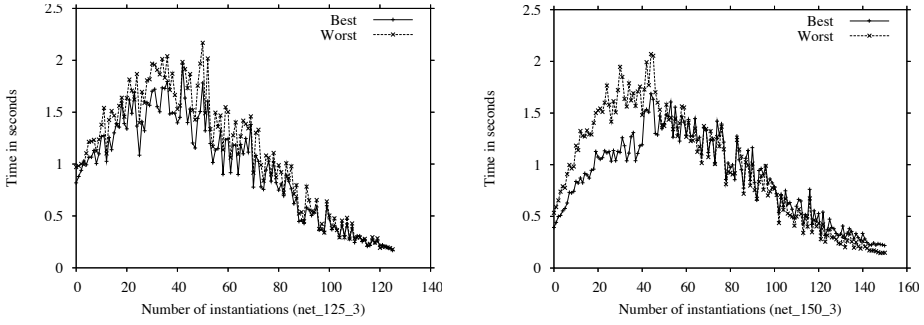

Fig. 6. Time savings of breaking ties by reversing the best and worst arcs as determined by the next cost measure in BreakTies on net125 (left) and net150 (right)

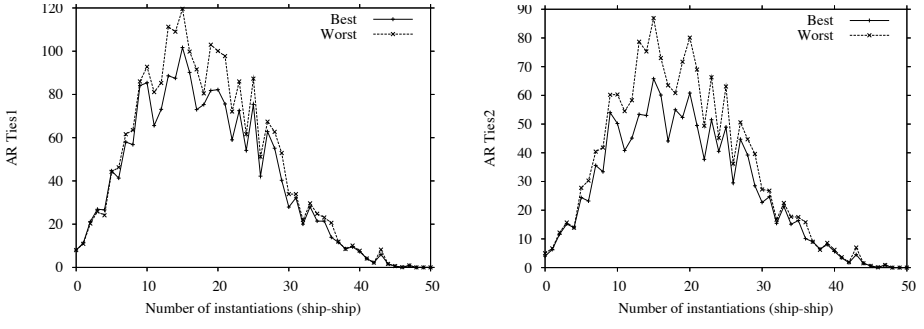

Fig. 7. Using BreakTies on ship-ship, the average number of *cptw* ties (left) and *cptw* and *fiw* ties (right)

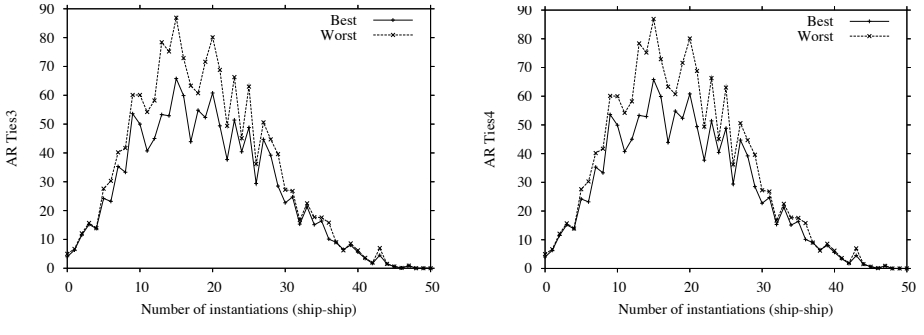

Fig. 8. Using BreakTies on ship-ship, the average number of *cptw, fiw*, and *fi* ties (left) and *cptw, fiw, fi*, and *nop* ties (right)

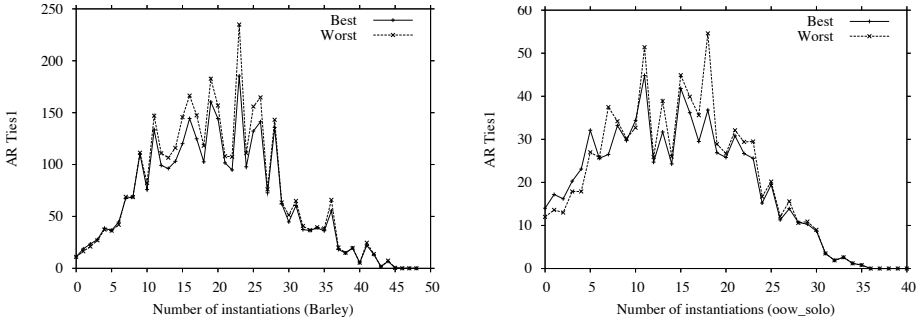

Fig. 9. The average number of *cptw* ties in BreakTies on Barley (left) and OOW (right)

operations when breaking ties, the average size of the largest CPT does not change noticeably.

The average number of *cptw* ties for ship-ship, Barley and OOW are shown in Figs. 7 (left) and 9, respectively. In the ship-ship Bayesian network, for instance, the average number of *fiw* ties, *fi* ties, and *nop* ties are obtained by subtracting (left) from (right) in Fig. 7, subtracting Fig. 7 (right) from Fig. 8 (left), and subtracting (left) from (right) in Fig. 8, respectively. The most significant differences in the average number of ties between selecting the *worst* and *best* arc to reverse for the four heuristics are achieved between ten and thirty evidence variables. Following the *best* heuristic offers a reduction in the average number of ties. For example, in the case of six evidence variables in Fig. 7 (left), the average number of ties was 41.3 using the *best* heuristic, while it was 46.3 using the *worst* heuristic. Similarly, for the case of six evidence variables in Fig. 7 (right) and Fig. 8 together, the average number of ties using the *best* and *worst* heuristics were 23.2 was 30.3, respectively.

6 Conclusions

When using cost measure *cptw* to reverse arcs in LPAR, [14] will reverse the first arc tied for the lowest score in the event of a tie. Instead, we suggest breaking ties with other cost measures *fiw*, *fi* and *nop*, in this order. Example 3 illustrates how random selection can result in more computation. The empirical experiments in Figs. 3 - 6 have produced two important findings. First, in most cases, breaking ties does not yield significant gains. Second, for both randomly generated and real-world networks, sometimes breaking ties produces a noticeable time improvement. The improvements are most significant for small subsets of evidence where the cost of inference is highest. The average number of ties encountered in BreakTies, reported in Figs. 7 - 9, illustrate that there are indeed ties to be broken. Future work will investigate combined cost measures.

Acknowledgments. This research is supported in part by NSERC Discovery Grant 238880.

References

1. Butz, C.J., Chen, J., Konkel, K., Lingras, P.: A Formal Comparison of Variable Elimination and Arc Reversal in Bayesian Network Inference. Intell. Dec. Analysis 3(3), 173–180 (2009)
2. Butz, C.J., Konkel, K., Lingras, P.: Join Tree Propagation Utilizing both Arc Reversal and Variable Elimination. Intl. J. Approx. Rea. (2010) (in press)
3. Butz, C.J., Hua, S., Konkel, K., Yao, H.: Join Tree Propagation with Prioritized Messages. Networks 55(4), 350–359 (2010)
4. Butz, C.J., Yao, H., Hua, S.: A Join Tree Probability Propagation Architecture for Semantic Modelling. J. Int. Info. Sys. 33(2), 145–178 (2009)
5. Cooper, G.F.: The Computational Complexity of Probabilistic Inference using Bayesian Belief Networks. Art. Intel. 42(2-3), 393–405 (1990)
6. Darwiche, A.: Modeling and Reasoning with Bayesian Networks. Cambridge University Press, New York (2009)
7. Hansen, P.F., Pedersen, P.T.: Risk Analysis of Conventional and Solo Watch Keeping. Research Report, Department of Naval Architecture and Offshore Engineering. Technical University of Denmark (1998)
8. Jensen, F.V., Nielsen, T.D.: Bayesian Networks and Decision Graphs. Springer, New York (2007)
9. Kjaerulff, U.B., Madsen, A.L.: Bayesian Networks and Influence Diagrams: a Guide to Construction and Analysis. Springer, New York (2008)
10. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. The MIT Press, Cambridge (2009)
11. Kristensen, K., Rasmussen, I.A.: The use of a Bayesian Network in the Design of a Decision Support System for Growing Malting Barley without use of Pesticides. Computers and Electronics in Agriculture 33, 192–217 (2002)
12. Madsen, A.L.: An Empirical Evaluation of Possible Variations of Lazy Propagation. In: 20th Conference in Uncertainty in Artificial Intelligence, pp. 366–373. AUAI Press, Arlington (2004)
13. Madsen, A.L.: Variations over the Message Computation Algorithm of Lazy Propagation. IEEE Trans. Sys. Man Cyb. B 36, 636–648 (2006)
14. Madsen, A.L.: Improvements to Message Computation in Lazy Propagation. Intl. J. Approx. Rea. 51(5), 499–514 (2010)
15. Madsen, A.L., Jensen, F.V.: A Junction Tree Inference Algorithm based on Lazy Evaluation. Art. Intel. 113(1-2), 203–245 (1999)
16. Neapolitan, R.E.: Probabilistic Methods for Bioinformatics with an Introduction to Bayesian Networks. Morgan Kaufmann, New York (2009)
17. Olmsted, S.: On Representing and Solving Decision Problems. Ph.D. Thesis, Department of Engineering Economic Systems. Stanford University, Stanford, California (1983)
18. Pearl, P.: Probabilistic Reasoning Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco (1988)
19. Pourret, O., Naim, P., Marcot, B. (eds.): Bayesian Networks: A Practical Guide to Applications. Wiley, New York (2008)
20. Shachter, R.: Evaluating Influence Diagrams. Oper. Research 34, 871–882 (1986)
21. Shafer, G.: Probabilistic Expert Systems. SIAM, Philadelphia (1996)