# An Improved LAZY-AR Approach to Bayesian Network Inference

C.J. Butz and S. Hua

Department of Computer Science,
University of Regina
Regina, SK, S4S 0A2, Canada
{butz, huash111}@cs.uregina.ca

**Abstract.** We propose LAZY arc-reversal with variable elimination (LAZY-ARVE) as a new approach to probabilistic inference in Bayesian networks (BNs). LAZY-ARVE is an improvement upon LAZY arc-reversal (LAZY-AR), which was very recently proposed and empirically shown to be the state-of-the-art method for exact inference in discrete BNs. The primary advantage of LAZY-ARVE over LAZY-AR is that the former only computes the actual distributions passed during inference, whereas the latter may perform unnecessary computation by constructing irrelevant intermediate distributions. A comparison between LAZY-AR and LAZY-ARVE, involving processing evidence in a real-world BN for coronary heart disease, is favourable towards LAZY-ARVE.

## 1 Introduction

*Bayesian networks* (BNs) [1, 2, 10, 14] are an established framework for uncertainty management in artificial intelligence. A BN consists of a *directed acyclic graph* (DAG) and a corresponding set of *conditional probability tables* (CPTs). The *probabilistic conditional independencies* (CIs) [15] encoded in the DAG indicate the product of CPTs is a joint probability distribution. Exact inference algorithms in BNs can be broadly classified into two categories. One approach is *join tree propagation* (JTP), which systematically passes messages in a *join tree* (JT) constructed from the DAG of a BN. The classical JTP methods were proposed by Lauritzen and Spiegelhalter [5], Shafer and Shenoy [14], and Jensen et al. [3]. Madsen and Jensen [7] suggested a JTP algorithm, called LAZY propagation, and empirically demonstrated a significant improvement in efficiency over the traditional JTP methods. A second approach to BN inference is *direct computation* (DC), which performs inference directly in a BN. The classical DC algorithms are variable elimination (VE) [17, 18, 19], arc-reversal (AR) [9, 13] and symbolic probabilistic inference (SPI) [6, 12]. The experimental results provided by Zhang [17] indicate that VE is more efficient than the classical JTP methods when updating twenty or less non-evidence variables, given a set of twenty or fewer evidence variables.

Very recently, Madsen [8] examined hybrid approaches to BN inference. Inference is still conducted in a JT, but DC computation is utilized to perform

L. Lamontagne and M. Marchand (Eds.): Canadian AI 2006, LNAI 4013, pp. 183–194, 2006.

the physical computation. Of the three hybrid approaches tested, LAZY arc-reversal (LAZY-AR) was empirically shown to be the state-of-the-art method for exact inference in discrete BNs [8]. When a JT node is ready to send its CPT messages to a neighbour, the LAZY-AR approach eliminates all variables not appearing in the neighbour node. In particular, eliminating variable $v$ may require directed edges (arcs) to be reversed in the DAG (defined by the CPTs at the sending JT node) in order to make $v$ *barren* [13]. Such arc reversals are useful, since it is well-known that barren variables can be exploited for more efficient inference [7]. The missing CPTs of the newly constructed DAG are physically built from the existing CPTs. We point out that the LAZY-AR approach is sometimes wasteful as it can construct intermediate CPTs that are immaterial.

In this paper, we propose LAZY arc-reversal with variable elimination (LAZY-ARVE) as a new approach to BN inference. As the name suggests, our method is based upon the LAZY-AR approach. Whereas LAZY-AR iterates between semantic modeling and physical computation, LAZY-ARVE performs semantic modeling and physical computation separately. More specifically, LAZY-ARVE first performs semantic modeling in order to identify those CPT messages to be sent to a neighbour JT node. LAZY-ARVE next physically constructs the distributions of the passed CPTs using the VE inference algorithm. There are important advantages to uncoupling the independent tasks of semantic modeling and physical computation. By treating these two tasks as dependent, LAZY-AR can construct intermediate CPTs that will neither be sent to a neighbour, nor needed in the construction of the propagated CPTs. Physically constructing these irrelevant intermediate CPTs not only wastes computation but also the time required to build these distributions. As the screen shot in Fig. 5 illustrates, we have implemented the AR approach to identify the CPTs to be propagated. Using a real-world BN for *coronary heart disease* (CHD) [2], we compared our approach of applying VE to build only the propagated CPTs with the state-of-the-art method. The results in Table 1, in which roughly eighteen percent of the BN variables are instantiated as evidence variables, show promise.

This paper is organized as follows. Section 2 contains background knowledge. In Section 3, we discuss a new approach to probabilistic inference. Related works are provided in Section 4. The conclusion is presented in Section 5.

## 2 Background Knowledge

Here we review Bayesian networks, probabilistic inference and the AR method.

### 2.1 Bayesian Networks

Let $U = \{v_1, v_2, \ldots, v_n\}$ denote a finite set of discrete random variables. Each variable $v_i$ is associated with a finite domain, denoted $dom(v_i)$, representing the values $v_i$ can take on. For a subset $X \subseteq U$, we write $dom(X)$ for the

Cartesian product of the domains of the individual variables in $X$. Each element $x \in dom(X)$ is called a *configuration* of X. A *potential* [2] on $dom(X)$ is a function $\phi$ on $dom(X)$ such that $\phi(x) \geq 0$, for each configuration $x \in dom(X)$, and at least one $\phi(x)$ is positive. For brevity, we refer to a potential as a probability distribution on $X$ rather than $dom(X)$, and we call $X$, not $dom(X)$, its domain [14]. A *joint probability distribution* (JPD) [14] on $U$, denoted $p(U)$, is a potential on $U$ that sums to one. Given $X \subset U$, a *conditional probability table* (CPT) [14] for a variable $v \notin X$ is a distribution, denoted $p(v|X)$, satisfying the following condition: for each configuration $x \in dom(X)$, $\sum_{c \in dom(v)} p(\ v = c \mid X = x\ ) = 1.0$.

A *Bayesian network* (BN) [10] on $U$ is a pair $(D, C)$. $D$ is a *directed acyclic graph* (DAG) on $U$. $C$ is a set of CPTs defined as: for each variable $v_i \in D$, there is a CPT for $v_i$ given its parents $P_i$ in $D$. Based on the *probabilistic conditional independencies* [15] encoded in $D$, the product of the CPTs in $C$ is a JPD $p(U)$.

*Example 1.* The DAG of one real-world BN for *coronary heart disease* (CHD) [2] is shown in Fig. 1. The corresponding CPTs are not pertinent to our discussion. For pedagogical reasons, we have made the following minor adjustments to the DAG: edge $(a, f)$ has been removed; edges $(c, f)$ and $(g, i)$ have been replaced with edges $(c, d)$, $(c, e)$, $(d, f)$, $(e, f)$ and $(g, j)$, where $d$ and $e$ are dummy variables.

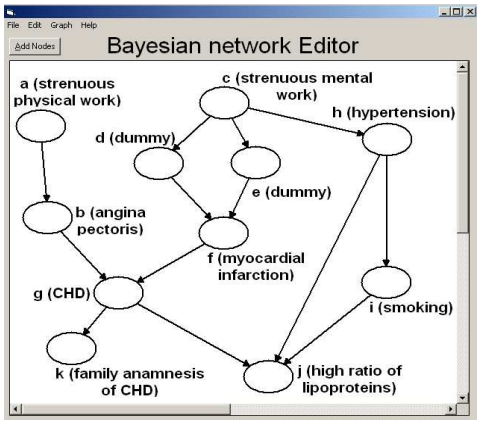


**Fig. 1.** The *coronary heart disease* (CHD) BN [2] in Example 1

We will use the terms BN and DAG interchangeably if no confusion arises. The *family* $F_i$ of a variable $v_i$ in a DAG is $\{v_i\} \cup P_i$. A numbering $\prec$ of the variables in a DAG is called *ancestral* [1], if the number corresponding to any variable $v_i$ is lower than the number corresponding to each of its children $v_j$, denoted $v_i \prec v_j$.

In the CHD BN in Fig. 1, we will always use the fixed ancestral numbering as $a \prec b \prec \ldots \prec k$.

### 2.2 Probabilistic Inference

In this paper, we only consider exact inference in discrete BNs. Probabilistic inference (or query processing) means computing $p(X)$ or $p(X|E = e)$, where $X$ and $E$ are disjoint subsets of $U$. The *evidence* in the latter query is that $E$ is instantiated to configuration $e$, while $X$ contains *target* variables. Barren variables can be exploited in inference [7]. A variable is *barren* [13], if it is neither an evidence nor a target variable and it only has barren descendants. Probabilistic inference can be conducted directly in the original BN [6, 9, 12, 13, 17, 18, 19]. It can also be performed in a join tree [3, 5, 7, 8, 14].

Shafer [14] emphasizes that *join tree propagation* (JTP) is central to the theory and practice of probabilistic expert systems. A *join tree* (JT) [10, 14] is a tree with sets of variables as nodes, with the property that any variable in two nodes is also in any node on the path between the two. The *separator* $S$ between any two neighbour nodes $N_i$ and $N_j$ is $S = N_i \cap N_j$. The task of transforming a DAG into a JT has been extensively studied in probabilistic reasoning literature. Note that constructing a minimal JT is NP-complete [16]. For example, recall the CHD BN in Fig. 1. One possible JT with nodes $\{ab, bfg, cdefgh, ghij, gk\}$ is depicted in Fig. 2 (ignoring the messages at the moment).

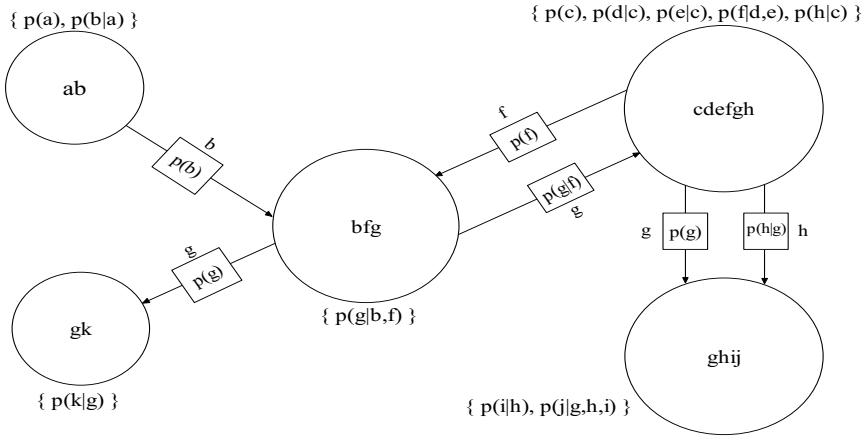


**Fig. 2.** A JT for the CHD BN in Fig. 1

Unlike traditional JTP approaches [3, 5, 14], LAZY propagation [7] maintains structure in the form of a multiplicative factorization of potentials at each JT node and each JT separator. Maintaining a decomposition of potentials offers LAZY the opportunity to exploit barren variables and independencies induced by

evidence. Doing so improves the efficiency of JTP remarkably as the experimental results in [7] clearly emphasize.

### 2.3 Arc Reversal

The basic idea of *Arc-reversal* (AR) [9, 13] is to make a variable barren via a sequence of arc reversals prior to eliminating it. Suppose variable $v_i$ is to be eliminated and arc $(v_i, v_j)$ needs to be reversed. The arc $(v_i, v_j)$ is graphically reversed as $(v_j, v_i)$ by setting the new parents of $v_i$ as $P_i \cup F_j - \{v_i\}$, while making $P_i \cup P_j - \{v_i\}$ the new parents of $v_j$. Note that AR uses an ancestral numbering $\prec$ of the given BN to avoid creating directed cycles. Hence, a DAG structure is maintained after eliminating a variable by applying AR [4, 8]. The next example illustrates how AR reverses arcs when eliminating variables during inference not involving evidence.

*Example 2.* Consider how node $cdefgh$ sends the CPT messages $\{p(g), p(h|g)\}$ to node $ghij$ in the JT in Fig. 2. Node $cdefgh$ collects the CPT $p(g|f)$ sent from $bfg$. A DAG in Fig. 3 (i) is defined on the set $C = \{p(c), p(d|c), p(e|c), p(f|d, e), p(g|f), p(h|c)\}$ of CPTs at $cdefgh$. Applying AR to eliminate variables $\{c, d, e, f\}$ gives the sub-DAG in Fig. 3 (ii). For pedagogical purposes, let us eliminate the variables in the order $d, c, e, f$. To eliminate $d$, arc $(d, f)$ needs to be reversed. Here, $v_i = d$, $P_i = \{c\}$, $v_j = f$, $P_j = \{d, e\}$ and $F_j = \{d, e, f\}$. The reversed arc $(f, d)$ is created by setting $P_i' = \{c, e, f\}$ and $P_j' = \{c, e\}$, as shown in Step 1 of Fig. 4 (i). Variable $d$ becomes barren and can be removed. The remaining sub-DAG under consideration is illustrated in Step 2 of Fig. 4 (i). For variable $c$, arcs $(c, e)$, $(c, f)$ and $(c, h)$ need to be reversed. According to $\prec$ of the CHD BN in Fig. 1, arc $(c, e)$ will be reversed first. Here, $v_i = c$, $P_i = \emptyset$, $v_j = e$, $P_j = \{c\}$ and $F_j = \{c, e\}$. The reversed arc $(e, c)$ is created by setting $P_i' = \{e\}$ and $P_j' = \emptyset$, as shown in Step 1 of Fig. 4 (ii). In a similar manner, arcs $(c, f)$ and $(c, h)$ are reversed as shown in Step 2 and Step 3 of Fig. 4 (ii), respectively. Variable $c$ becomes barren and can be removed giving the sub-DAG in Step 4

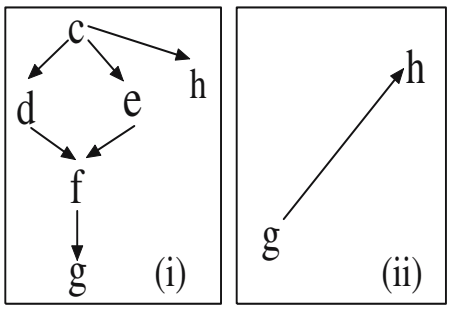


**Fig. 3.** (i) The initial DAG for Example 2. (ii) Applying AR to eliminate $\{c, d, e, f\}$ yields this sub-DAG.

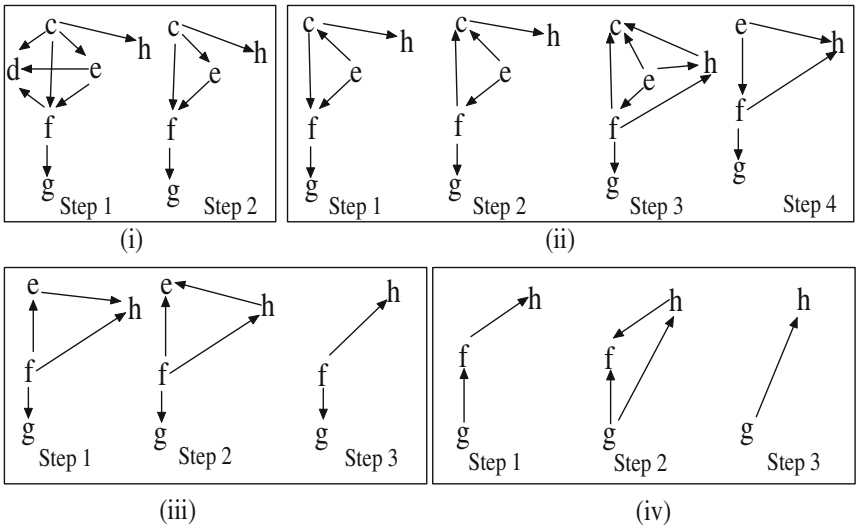


**Fig. 4.** Illustrating Arc-Reversal (AR) by eliminating variables $d$ (i), $c$ (ii), $e$ (iii) and $f$ (iv) from the initial DAG in Fig. 3 (i). The final DAG is shown in Fig. 3 (ii).

of Fig. 4 (ii). Similarly, variables $e$ and $f$ are removed via the arc reversals as shown in Fig. 4 (iii) and (iv), respectively.

## 3 New Approach LAZY-ARVE for BN Inference

In this section, we introduce LAZY Arc-Reversal with Variable Elimination (LAZY-ARVE) as a new algorithm for BN inference. LAZY-ARVE is built upon the AR Message Identification (ARMI) and VE [17, 18, 19] algorithms. First, the sub-algorithm ARMI applies AR for the graphical identification of the propagated CPTs. Next, VE is applied to compute only the propagated CPTs.

**Algorithm 1.** LAZY-ARVE($C$, $X$)
Input: a set $C$ of CPT distributions at a JT node,
the set $X$ of variables to be eliminated from $C$.
Output: the set $C$ of CPT distributions sent from a JT node to a neighbour.
**begin**
1. Call ARMI to identify the labels of the propagated CPTs.
2. **for** each CPT label $p(v|X)$ output from Step 1
Call VE to physically construct the actual distribution.
3. **return** the physical distributions output from Step 2
**end**

The sub-algorithm ARMI applies AR to *identify* the labels of the actual CPT messages to be sent from a JT node to a neighbour.

**Algorithm 2.** ARMI ($C$,$X$,$\prec$)
Input: a set $C$ of CPT labels at a JT node,
the set $X$ of variables to be eliminated from $C$,
an ancestral numbering $\prec$ of the variables in the BN.
Output: the set $C$ of CPT labels sent from a JT node to a neighbour.
**begin**
Construct the DAG $G = (V, E)$ defined by $C$.
**for** each variable $v_i$ in $X$
{
Let $Y = \{v_1, \ldots, v_k\}$ be the set of all children of $v_i$ in $G$, where $v_1 \prec \ldots \prec v_k$.
**for** $j = 1, \ldots, k$
{
$P'_i = P_i \cup F_j - \{v_i\}$
$P'_j = P_i \cup P_j - \{v_i\}$
$C = C \cup \{ p(v_i|P'_i), p(v_j|P'_j) \} - \{ p(v_i|P_i), p(v_j|P_j) \}$
}
Remove barren variable $v_i$ from $G$ and its CPT from $C$.
}
**return**($C$)
**end**

After identifying the CPT labels to be sent from a JT node, the VE [17, 18, 19] algorithm is called to physically *compute* the propagated CPTs. To compute $p(X|Y = Y_0)$, VE calls the sub-algorithm sum-out to eliminate variables outside $X \cup Y$ from a list of factors one by one, according to an elimination ordering $\sigma$.

**Algorithm 3.** [19] VE($\mathcal{F}$, $X$, $Y$, $Y_0$, $\sigma$)
Input: $\mathcal{F}$ - the list of conditional probabilities in a BN,
$X$ - a list of query variables,
$Y$ - a list of observed variables,
$Y_0$ - the corresponding list of observed values,
$\sigma$ - an elimination ordering for variables outside $X \cup Y$.
Output: the physical distribution of $p(X|Y = Y_0)$.
**begin**
Set the observed variables in all factors to their corresponding observed values.
**While** $\sigma$ is not empty,
{
Remove the first variable $z$ from $\sigma$,
$\mathcal{F}$ = sum-out($\mathcal{F}$,$z$).
}
Set $h$ = the multiplication of all the factors on $\mathcal{F}$.
Return $p(X|Y = Y_0) = h(X)/\sum_X h(X)$.
**end**

**Algorithm 4.** [19] sum-out($z$,$\mathcal{F}$)
Input: $\mathcal{F}$ - a list of factors,
$z$ - a variable.
Output: another list of factors.

**begin**
Remove from $\mathcal{F}$ all the factors, say $f_1, \ldots, f_k$, that contain $z$.
Add the new factor $\sum_z \prod_{i=1}^{k} f_i$ to $\mathcal{F}$.
Return $\mathcal{F}$.
**end**

The next example illustrates our LAZY-ARVE inference algorithm.

*Example 3.* Recall Fig. 3 (i). In Step 1, ARMI returns the CPT labels $\{p(g), p(h|g)\}$, as depicted in Fig. 3 (ii), to be sent from node $cdefgh$ to $ghij$ in the JT in Fig. 2. In Step 2, LAZY-ARVE calls VE to compute the physical distributions for $p(g)$ and $p(h|g)$.

Consider physically computing $p(h|g)$. Here, $\mathcal{F} = \{p(c), p(d|c), p(e|c), p(f|d,e), p(g|f), p(h|c)\}$, $X = h$, $Y = g$ and $Y_0 = \emptyset$. Suppose the elimination ordering $\sigma$ is $\{d, c, e, f\}$. The first variable $d$ in $\sigma$ is removed by calling sum-out, which computes $\phi(c,e,f) = \sum_d p(d|c) \cdot p(f|d,e)$ and then sets $\mathcal{F} = \{p(c), \phi(c,e,f), p(e|c), p(g|f), p(h|c)\}$. Next, sum-out removes variable $c$ as $\phi(e,f,h) = \sum_c p(c) \cdot \phi(c,e,f) \cdot p(e|c) \cdot p(h|c)$ leaving $\mathcal{F} = \{\phi(e,f,h), p(g|f)\}$. It can be verified that $\mathcal{F} = \phi(g,h)$ after removing $\{e,f\}$. Lastly, VE computes $p(h|g)$ by normalizing $\phi(g,h)$, i.e., $p(h|g) = \phi(g,h)/\sum_h \phi(g,h)$. The CPT $p(g)$ can be similarly constructed.

The important point to remember is that LAZY-ARVE uses AR to maintain CPT structure during the elimination of variables, yet applies VE to physically compute only those CPTs propagated in the JT.

## 4 Related Works

This section compares the computation work required by LAZY-ARVE with LAZY-AR, the state-of-the-art algorithm recently proposed by Madsen [8] for exact inference in discrete BNs. Our comparison is conducted in the real-world CHD BN of Fig. 1. Since the work for identification of the passed CPTs is the same for both algorithms, we only contrast the physical computation.

LAZY-AR implements arc-reversal (AR) as the engine for performing inference in LAZY propagation with impressive experimental results [8]. Roughly speaking, LAZY-AR maintains a sub-BN at a JT node after eliminating a variable by applying AR and constructing the corresponding CPTs of the sub-BN. The following outline draws from [4, 8]. Let variable $v_i$ be eliminated and arc $(v_i, v_j)$ needs to be reversed. Assume $v_i$ has parents $P_i = X_m \cup X_n$ and variable $v_j$ has parents $P_j = \{v_i\} \cup X_n \cup X_k$, where $X_m \cap X_n = X_m \cap X_k = X_n \cap X_k = \emptyset$ such that $X_m = P_i - P_j$ are the parents of $v_i$ but not $v_j$, $X_n = P_i \cap P_j$ are parents of both $v_i$ and $v_j$, and $X_k = P_j - F_i$ are the parents of $v_j$ but not $v_i$ and its parents. Arc $(v_i, v_j)$ is reversed by setting $P_i = X_m \cup X_n \cup X_k \cup \{v_j\}$ and $P_j = X_m \cup X_n \cup X_k$. Next, new CPTs for $v_i$ and $v_j$ in the modified DAG are physically constructed as follows:

$$p(v_j \mid X_m, X_n, X_k) = \sum_{v_i} p(v_i \mid X_m, X_n) \cdot p(v_j \mid v_i, X_n, X_k), \qquad (1)$$

$$p(v_i \mid v_j, X_m, X_n, X_k) \;=\; \frac{p(v_i \mid X_m, X_n) \cdot p(v_j \mid v_i, X_n, X_k)}{p(v_j \mid X_m, X_n, X_k)} \;. \tag{2}$$

Note that it is not necessary to perform the last invocation of Equation (2) when the last arc from $v_i$ is reversed, since $v_i$ will be eliminated as a barren variable.

The next example shows how LAZY-AR physically constructs a sequence of intermediate CPTs by applying Equations (1) and (2).

*Example 4.* Recall Example 2. Let us show the physical computation performed by LAZY-AR to construct the propagated CPTs $p(g)$ and $p(h|g)$ from the JT node $cdefgh$ to the JT node $ghij$. In Step 1 of Fig. 4 (i), the two new CPTs are $p(d|c,e,f)$ and $p(f|c,e)$. While the former need not be computed as $d$ is barren, the latter is constructed by Equation (1) as:

$$p(f|c,e) \;=\; \sum_d p(f|d,e) \cdot p(d|c) \;.$$

In Step 4 of Fig. 4 (ii), computing three new CPTs $p(e)$, $p(f|e)$ and $p(h|e,f)$ after elimination of $c$ requires the construction of five intermediate CPTs:

$$\begin{aligned}
p(e) &= \sum_c p(c) \cdot p(e|c), \\
p(c|e) &= p(c) \cdot p(e|c)/p(e), \\
p(f|e) &= \sum_c p(c|e) \cdot p(f|c,e), \\
p(c|e,f) &= p(c|e) \cdot p(f|c,e)/p(f|e), \\
p(h|e,f) &= \sum_c p(c|e,f) \cdot p(h|c).
\end{aligned}$$

To eliminate $e$ and $f$, LAZY-AR needs to construct six intermediate CPTs:

$$p(f) \;=\; \sum_e p(e) \cdot p(f|e), \tag{3}$$
$$p(e|f) \;=\; p(e) \cdot p(f|e)/p(f), \tag{4}$$
$$p(h|f) \;=\; \sum_e p(e|f) \cdot p(h|e,f), \tag{5}$$
$$p(g) \;=\; \sum_f p(f) \cdot p(g|f), \tag{6}$$
$$p(f|g) \;=\; p(f) \cdot p(g|f)/p(g), \tag{7}$$
$$p(h|g) \;=\; \sum_f p(f|g) \cdot p(h|f) \;. \tag{8}$$

The next example illustrates that LAZY-AR physically constructs intermediate CPTs that will neither be passed during inference, nor required in the construction of those CPTs actually passed in the JT.

*Example 5.* In Example 4, after variables $d$ and $c$ are removed, variable $e$ can be simply eliminated as follows:

$$\sum_e p(e) \cdot p(f|e) \cdot p(h|e,f). \tag{9}$$

On the contrary, by reversing $(e, f)$ as $(f, e)$, LAZY-AR eliminates $e$ as:

$$\sum_e p(f) \cdot p(e|f) \cdot p(h|e,f), \tag{10}$$

which requires the physical construction of $p(f)$ and $p(e|f)$ in Equations (3) and (4), respectively. By substituting Equation (3) into Equation (10), we obtain

$$\sum_e \ [ \ \sum_e p(e) \cdot p(f|e) \ ] \cdot p(e|f) \cdot p(h|e,f). \tag{11}$$

By substituting Equation (4) into Equation (11), we have

$$\sum_e \ [ \ \sum_e p(e) \cdot p(f|e) \ ] \cdot \frac{p(e) \cdot p(f|e)}{p(f)} \cdot p(h|e,f). \tag{12}$$

By Equation (3), we can rewrite Equation (12) as

$$\sum_e p(e) \cdot p(f|e) \cdot p(h|e,f) \cdot \frac{\sum_e p(e) \cdot p(f|e)}{\sum_e p(e) \cdot p(f|e)}. \tag{13}$$

During its physical computation, Equation (13) indicates that LAZY-AR multiplies and divides the same term

$$\textstyle\sum_e p(e) \cdot p(f|e).$$

By comparing Equations (9) and (13), it is explicitly demonstrated that LAZY-AR performs unnecessary computation by physically constructing intermediate CPTs that will neither be passed during inference, nor required in the construction of the actual propagated CPTs. Although intermediate CPTs are useful for message identification, they are not necessarily needed for message construction. Any redundant work will delay the construction of the actual CPTs required for BN inference.

Similar to the comparisons made by Schmidt and Shenoy [11], we conclude this section by providing the following comparison between LAZY-AR and LAZY-ARVE. Approximately eighteen percent of the CHD BN variables are instantiated as evidence variables, such as was done for the largest BN used in the experimental results of [7].

*Example 6.* Given $b = 0$ and $g = 0$ as collected evidence in the CHD JT in Fig. 2. The screen shot of our implemented system in Fig. 5 shows all identified CPT messages. Table 1 shows the work needed by our LAZY-ARVE approach and LAZY-AR to physically construct the CPT messages $p(b = 0)$, $p(f)$, $p(g = 0|b = 0)$, $p(g = 0|b = 0, f)$, and $p(h|b = 0, g = 0)$ in Fig. 5.

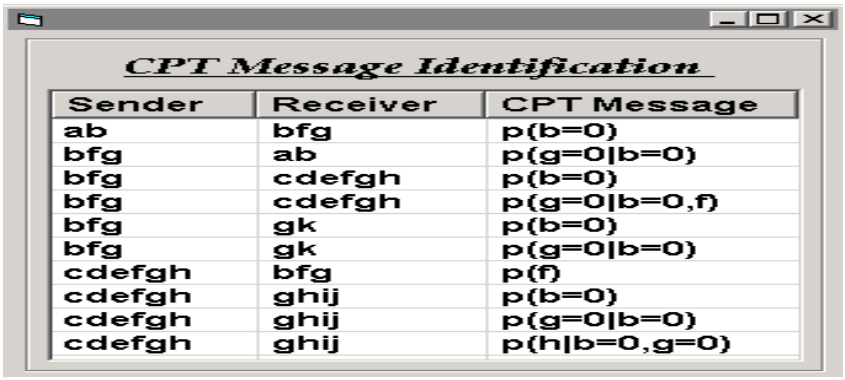


**Fig. 5.** In the CHD JT in Fig. 2, given evidence $b = 0$ and $g = 0$, our implemented system identifies the CPT messages to be propagated

**Table 1.** Given evidence $b = 0$ and $g = 0$, the computation needed in LAZY-AR versus LAZY-ARVE to physically construct the CPTs passed in the CHD JT

| CPT | LAZY-AR | | | LAZY-ARVE | | |
|---|---|---|---|---|---|---|
| message | $+$ | $\times$ | $\div$ | $+$ | $\times$ | $\div$ |
| $p(b = 0)$ | 1 | 2 | 0 | 1 | 2 | 0 |
| $p(f)$ | 14 | 36 | 8 | 11 | 20 | 2 |
| $p(g = 0\|b = 0)$ | 13 | 34 | 8 | 11 | 24 | 1 |
| $p(g = 0\|b = 0, f)$ | 0 | 0 | 0 | 0 | 0 | 0 |
| $p(h\|b = 0, g = 0)$ | 29 | 80 | 22 | 19 | 44 | 2 |

The results in Table 1 suggest that our LAZY-ARVE has promise. In fact, the LAZY-ARVE can be fine-tuned by re-using some calculations. For instance, some work required to build $p(h|g)$ in Example 3 can be utilized when computing $p(g)$. Formal experimental results consisting of running times for exact inference in large, discrete, real-world BNs will be presented in a forthcoming paper.

## 5 Conclusions

In this paper, we propose LAZY-ARVE as a new approach to probabilistic inference in BNs. LAZY-ARVE is an improvement upon LAZY-AR, which was very recently proposed and empirically shown to be the state-of-the-art method for exact inference in discrete BNs [8]. However, intermediate CPTs computed by LAZY-AR may be irrelevant to BN inference. The reason for this unnecessary computation is that LAZY-AR iterates between semantic modeling and physical computation. Although intermediate CPT labels are useful for semantic modeling, the corresponding distributions do not necessarily have to be physically computed. We suggest separating these two independent tasks. Semantic computation is carried out first by implementing AR only to graphically identify the

CPTs passed between JT nodes. Next, the VE [17, 18, 19] inference algorithm is applied to physically construct the distributions of the propagated CPTs. Table 1 seems to imply that LAZY-ARVE could be the state-of-the-art algorithm for exact probabilistic inference in discrete BNs. Formal experimental results will be presented shortly.

## References

1. E. Castillo, J. Gutiérrez and A. Hadi, Expert Systems and Probabilistic Network Models, Springer, New York, 1997.
2. P. Hájek, T. Havránek and R. Jiroušek, Uncertain Information Processing in Expert Systems, CRC Press, Ann Arbor, 1992.
3. F.V. Jensen, S.L. Lauritzen and K.G. Olesen, Bayesian updating in causal probabilistic networks by local computations, Comp. St. Q., 4, pp. 269-282, 1990.
4. U. Kjaeulff and A.L. Madsen, Probabilistic Networks - An Introduction to Bayesian Networks and Influence Diagrams, 133 pages, Unpublished, 2005.
5. S.L. Lauritzen and D.J. Spiegelhalter, Local computations with probabilities on graphical structures and their application to expert systems, J. Roy. Statistical Society, B, 50(2), pp. 157-224, 1988.
6. Z. Li and B. D'Ambrosio, Efficient inference in Bayes networks as a combinatorial optimization problem, Internat. J. Approx. Reason., 11(1), pp. 55-81, 1994.
7. A.L. Madsen and F.V. Jensen, LAZY propagation: A junction tree inference algorithm based on lazy evaluation, Artif. Intell., 113 (1-2), pp. 203-245, 1999.
8. A.L. Madsen, An empirical evaluation of possible variations of lazy propagation, in: Proc. 20th Conference on Uncertainty in Artificial Intelligence, 2004, pp. 366-373.
9. S. Olmsted, On representing and solving decision problems, Ph.D. thesis, Department of Engineering Economic Systems, Stanford University, Stanford, CA., 1983.
10. J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.
11. T. Schmidt and P.P. Shenoy, Some improvements to the Shenoy-Shafer and Hugin architectures for computing marginals, Artif. Intell., 102, pp. 323-333, 1998.
12. R. Shachter, B. D'Ambrosio and B. Del Favero, Symbolic probabilistic inference in belief networks, in Proc. 8th National Conference on Artificial Intelligence, 1990, pp. 126-131.
13. R. Shachter, Evaluating inflence diagrams, Oper. Res., 34(6), pp. 871-882, 1986.
14. G. Shafer, Probabilistic Expert Systems, Society for Industrial and Applied Mathematics, 1996.
15. S.K.M. Wong, C.J. Butz and D. Wu, On the implication problem for probabilistic conditional independency, IEEE Trans. Syst. Man Cybern., A, 30(6), pp. 785-805, 2000.
16. M. Yannakakis, Computing the minimal fill-in is NP-complete, SIAM J. of Algebraic Discrete Methods, 2, pp. 77-79, 1981.
17. N.L. Zhang, Computational properties of two exact algorithms for Bayesian networks, Appl. Intell., 9 (2), pp. 173–184, 1998.
18. N.L. Zhang and D. Poole, A simple approach to Bayesian network computations, In: Proc. 10th Canadian Conference on Artificial Intelligence, 1994, pp. 171-178.
19. N.L. Zhang and D. Poole, Exploiting causal independence in Bayesian network inference, J. Artif. Intell. Res., 5, pp. 301-328, 1996.