Demystify the Messages in the Hugin Architecture for Probabilistic Inference and Its Application

Dan Wu
School of Computer Science
University of Windsor
Windsor Ontario
Canada N9B 3P4

Karen Jin
School of Computer Science
University of Windsor
Windsor, Ontario
Canada N9B 3P4

Abstract

In this paper, we investigate the semantic meaning of the messages passed in the Hugin architecture for probabilistic inference. By utilizing this information, one can avoid passing up to half of the messages that could have had to be passed in the Hugin architecture.

1. Introduction

The *global propagation* (GP) (Huang & Darwiche 1996) method used in the Hugin architecture (Lepar & Shenoy 1998) is arguably one of the best methods for probabilistic inference in Bayesian networks. Passing messages between cliques in a junction tree is the basic operation in the GP method. It is traditionally considered that the messages passed are simply potentials without any specific semantic meaning. No research has been reported on studying the algebraic properties of the messages.

In this paper, by studying the factorizations of a joint probability distribution defined by a Bayesian network *before* and *after* the GP method is performed, we investigate the messages passed algebraically, and we make the following two contributions. (a) We reveal that the messages passed are not mere potentials, but in fact separator marginals or factors in their factorizations. (b) We demonstrate that the revealed semantics of the messages can be utilized to avoid passing up to half of the messages that could have had to be passed by the GP method.

The paper is organized as follows. We present background knowledge in Section 2. In Section 3, we study different factorizations of the joint probability distribution defined by a Bayesian network. By comparing these factorizations, the semantics of messages passed in the GP method is gradually revealed. In Section 4, we use an example to show that utilizing the semantics of the messages revealed is potentially more efficient than the GP method in the Hugin architecture. We discuss future work and conclude the paper in Section 5.

2. Background

We use a upper case letter (possibly with a subscript) to represent a set of discrete variables and a lower case letter (possibly with a subscript) to represent one discrete variable (or a singleton set). The juxtaposition of two or more letters (either uppercase or lowercase) represents the union of the sets denoted by the letters. We use $p(X)$ to denote a *joint probability distribution* (JPD) over a set X of variables, and we call $p(Y)$ where $Y \subseteq X$ a *marginal* distribution (of $p(X)$). We use $p(X|Y)$ to denote the *conditional probability distribution*(CPD) of X given Y, and X is the *head* and Y is the *tail* of this CPD. A marginal $p(X)$ can also be considered as a CPD with its tail empty (i.e., $p(X|\emptyset)$). We use $I(X, Y, Z)$ to denote that X and Z are conditional independent (CI) given Y (Pearl 1988). A *potential* over a set X of variables, denoted ϕ_X or $\phi_X(X)$, is a non-negative function.

Traditionally, a *Bayesian network* (BN) defined over a set $V = \{a_1, \ldots, a_n\}$ of variables is a *directed acyclic graph* (DAG) augmented with a set of CPDs. More precisely, each variable a_i in V is represented as a node in the DAG and is associated with a CPD $p(a_i|\pi_{a_i})$, where π_{a_i} denotes the parents of a_i in the DAG. The product of these CPDs defines a JPD as:

$$p(V) = \prod_{a_i \in V} p(a_i|\pi_{a_i}), \tag{1}$$

and we call this factorization (in terms of CPDs) a *Bayesian factorization*. It is important to note that each variable $a_i \in V$ appears *exactly once* as the head of one CPD in the Bayesian factorization.

Alternatively and equivalently, a BN can also be defined in terms of the CPD factorization of a JPD.

Definition 1 Let $V = \{a_1, \ldots, a_n\}$. Consider the CPD factorization below:

$$p(V) = \prod_{a_i \in V,\ a_i \notin A_i,\ A_i \subseteq V} p(a_i|A_i), \tag{2}$$

If (1) each $a_i \in V$ appears exactly once as the head of one CPD in the above factorization, and (2) the graph obtained by depicting a directed edge from b to a_i for each $b \in A_i$ is a DAG, $i = 1, \ldots, n,$, then the DAG drawn and the CPDs $p(a_i|A_i)$ in Eq. (2) define a BN. In fact, the factorization in Eq. (2) is a Bayesian factorization of the defined BN.

An ordering of *all* the variables in a DAG is called a *topological ordering* if for each variable a_i in the DAG, the variables in π_{a_i} precede a_i in the ordering. We generalize the notion to a subset of variables in a DAG. Let V represent the set of all variables in a DAG, a subset $X \subseteq V$ of variables is said to be in a topological ordering with respect to the DAG, if for each variable $a_i \in X$, the variables in the intersection of a_i's ancestors and X precede a_i in the ordering. We say that a Bayesian factorization follows a topological ordering if for each CPD $p(a_i|\pi_{a_i})$, variables in π_{a_i} precede variable a_i in the ordering. Note that the DAG of a BN may have many different topological orderings, and its Bayesian factorization follows all these orderings. We can say that all these topological orderings lead to the same Bayesian factorization (Pearl 1988).

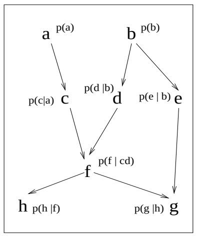

Figure 1: The DAG of the Asia travel BN.

Example 1 Consider the Asia travel BN defined over $V = \{a, \ \ldots, \ h\}$ from (Lauritzen & Spiegelhalter 1988). Its DAG and the CPDs associated with each node are depicted in Fig. 1. The JPD $p(V)$ is obtained as: $p(V) = p(a) \cdot p(b) \cdot p(c|a) \cdot p(d|b) \cdot p(e|b) \cdot p(f|cd) \cdot p(g|ef) \cdot p(h|f)$. The ordering $< a, \ b, \ c, \ d, \ e, \ f, \ g, \ h >$ is a topological ordering. The ordering $< b, \ d, \ h >$ is a topological ordering for the subset $\{b, \ d, \ h\}$ with respect to the DAG.

A BN is normally transformed into a junction tree for probabilistic inference. Among various algorithms developed, the GP method (Lauritzen & Spiegelhalter 1988; Shafer 1991; Jensen, Lauritzen, & Olesen 1990; Lepar & Shenoy 1998) is well received and implemented, for instance, in the renowned Hugin architecture. The GP method realizes the inference task not directly on the DAG of a BN, but on a secondary structure called *junction tree*. The junction tree is constructed from the DAG through moralization and triangulation. The GP method in essence is a coordinated series of local manipulations called *message passes* on the junction tree. Readers are referred to (Huang & Darwiche 1996) for detailed exposition. The following highlights pertinent facts of the GP method that are relevant to the discussions in this paper using an example.

Consider the Asia travel BN in Example 1. The DAG in Fig. 1 is moralized and triangulated so that a junction tree such as the one in Fig. 2 (i) is constructed. This junction tree consists of 6 cliques depicted as round rectangles, denoted $c_1 = ac$, $c_2 = bde$, $c_3 = cdf$, $c_4 = def$, $c_5 = fh$, $c_6 = efg$, and 5 separators depicted as smaller rectangles attached to the edge connecting two incidental cliques, denoted $s_1 = c$, $s_2 = de$, $s_3 = df$, $s_4 = f$, $s_5 = ef$.

Every CPD $p(a_i|\pi_{a_i})$ in Fig. 1 is assigned to a clique c_j if $\{a_i\} \cup \pi_{a_i} \subseteq c_j$ to form the clique potential ϕ_{c_j} before the GP method is applied. If $\{a_i\} \cup \pi_{a_i}$ is a subset of two

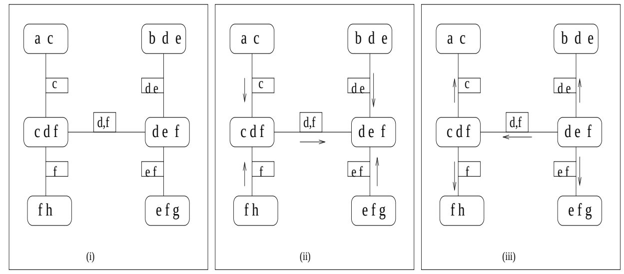

Figure 2: (i) The Junction tree constructed from the DAG in Fig. 1. (ii) Passing messages in the Collect-Evidence stage, and (iii) passing message in the Distribute-Evidence stage, when $c_4 = def$ is chosen as the root.

or more cliques, then we arbitrarily assign $p(a_i|\pi_{a_i})$ to one of the cliques. If no CPD is assigned to a clique c_j, then $\phi_{c_j} = 1$. In our example, the following clique potentials will be obtained before the GP is applied:

$$\begin{aligned} \phi_{c_1}(ac) &= p(a) \cdot p(c|a), & \phi_{c_4}(def) &= 1 \\ \phi_{c_2}(bde) &= p(b) \cdot p(d|e) \cdot p(e|b), & \phi_{c_5}(fh) &= p(h|f) \\ \phi_{c_3}(cdf) &= p(f|cd), & \phi_{c_6}(efg) &= p(g|fe) \end{aligned} \tag{3}$$

In the meantime, a separator potential is also formed for each separator with initial value 1, that is, $\phi_{s_i} = 1, i = 1, \ \ldots, \ 5$. It is known that the following equation holds before the GP method is performed on the junction tree:

$$p(V) \quad = \quad \phi_{c_1} \cdot \phi_{c_2} \cdot \phi_{c_3} \cdot \phi_{c_4} \cdot \phi_{c_5} \cdot \phi_{c_6}. \tag{4}$$

Note the separator potentials $\phi_{s_i}(\cdot)$ are identity potential ($\phi_{s_i} = 1$) before the GP method is applied.

The basic operation in the GP method is a local computation called *message pass*. Consider two adjacent cliques c_i and c_j with the separator s_{ij}, that c_i *passes a message* to c_j (or c_j *absorbs* the message from c_i) means a two-step computation: (1) updating the separator clique $\phi_{s_{ij}}$ by setting $\phi_{s_{ij}} = (\sum_{c_i - s_{ij}} \phi_{c_i})/\phi_{s_{ij}}$; (2) updating the clique potential ϕ_{c_j} by setting $\phi_{c_j} = \phi_{c_j} \cdot \phi_{s_{ij}}$. The potential $\phi_{s_{ij}}$ is the so-called "message" passed from c_i to c_j. Obviously, $\phi_{s_{ij}}$ in general is just a non-negative function.

The GP method is a coordinated sequence of message passes. Consider a junction tree with n cliques. It begins by picking any clique in the junction tree as the root, and then performs a sequence of message passes divided into two stages, namely, the *Collect-Evidence* stage, and the *Distribute-Evidence* stage. During the *Collect-Evidence* pass, each clique in the junction tree passes a message to its neighbors towards the root, beginning with the clique farthest from the root. During the *Distribute-Evidence* pass, each clique in the junction tree passes a message to its neighbor away from the root's direction, beginning with the root itself. The *Collect-Evidence* stage causes $n - 1$ messages to be passed. Similarly, the *Distribute-Evidence* stage causes another $n - 1$ messages to be passed. Altogether, there are exact $2(n - 1)$ messages to be passed (Huang & Darwiche 1996; Jensen 1996). The sequence of message passes is shown in Fig. 2 (ii) and (iii) when the clique $c_4 = def$ was chosen as the root.

After passing all $2(n-1)$ messages, the potentials ϕ_{c_i} and ϕ_{s_j} will have been turned into marginals $p(c_i)$ and $p(s_j)$ respectively, and the following equation holds (Huang & Darwiche 1996).

$$p(V) = \frac{p(c_1) \cdot p(c_2) \cdot p(c_3) \cdot p(c_4) \cdot p(c_5) \cdot p(c_6)}{p(s_1) \cdot p(s_2) \cdot p(s_3) \cdot p(s_4) \cdot p(s_5)}. \quad (5)$$

Although the mechanism of the GP method is well understood, what indeed the semantic meaning of the messages ϕ_{s_i} is remains mysterious.

3. Demystify the Messages

3.1 A Motivating Example

Comparing Eq. (4) with Eq. (5) leads us to the demystification of the messages. In the following, we will consider how one can transform Eq. (4) to Eq. (5) *algebraically*.

Recall that the clique potentials in Eq. (4) are in fact composed of the original CPDs from the BN shown in Fig. 1. If we substitute the actual contents for the clique potentials in Eq. (4), we obtain the following:

$$p(V) = \overbrace{[p(a) \cdot p(c|a)]}^{\phi_{c_1}} \cdot \overbrace{[p(b) \cdot p(d|e) \cdot p(e|b)]}^{\phi_{c_2}} \cdot \overbrace{[p(f|cd)]}^{\phi_{c_3}} \cdot \overbrace{[1]}^{\phi_{c_4}} \cdot \overbrace{[p(h|f)]}^{\phi_{c_5}} \cdot \overbrace{[p(g|fe)]}^{\phi_{c_6}} \quad (6)$$

Comparing Eq. (6) with Eq. (5), one may immediately notice that Eq. (6) does not have any denominators as Eq. (5) does. By multiplying and dividing $\Pi_{j=1}^{5} p(s_j)$ at the same time to Eq. (6), one will obtain[1]:

$$p(V) = [c \cdot de \cdot df \cdot f \cdot ef] \cdot \frac{\overbrace{[a, c|a]}^{c_1} \cdot \overbrace{[b, d|e, e|b]}^{c_2} \cdot \overbrace{[f|cd]}^{c_3} \cdot \overbrace{[1]}^{c_4} \cdot \overbrace{[h|f]}^{c_5} \cdot \overbrace{[g|fe]}^{c_6}}{c \cdot de \cdot df \cdot f \cdot ef}. \quad (7)$$

Comparing Eq. (7) with Eq. (5), one may find that they both have exactly the same denominators except the numerators. In Eq. (7), we now have some extra dangling marginals, namely $p(s_j)$, that are acquired when we multiply $\Pi_{j=1}^{5} p(s_j)$ to Eq. (6), and we hope that by mingling these extra marginals appropriately with the existing CPDs in the numerators in Eq. (7), we can reach Eq. (5).

With the ultimate goal of transforming the product in each square bracket (namely, the clique potential) in Eq. (7) into a clique marginal on its respective clique in mind, we examine how each separator marginal multiplied can be allocated to appropriate clique potentials in Eq. (3). It is obvious that: $\phi_{c_1}(ac) = p(a) \cdot p(c|a) = p(ac)$ and $\phi_{c_2}(bde) = p(b) \cdot p(d|b) \cdot p(e|b) = p(bde)$. In other words, no separator marginal should be mingled with these two potentials to make them marginals. For the clique potential $\phi_{c_5}(fh) = p(h|f)$, we can multiply it with the separator marginal $p(f)$ which results in $\phi_{c_5}(fh) = p(h|f) \cdot p(f) = p(fh)$. For

[1]Due to limited space, we write a for $p(a)$, $b|d$ for $p(b|d)$, etc.

potential	receives	result
ϕ_{c_1}	nothing	$p(a) \cdot p(c\|a) = p(ac)$
ϕ_{c_2}	nothing	$p(b) \cdot p(d\|b) \cdot p(e\|b) = p(bde)$
ϕ_{c_3}	$\underline{p(c)}, \underline{p(d)}$	$p(f\|cd) \cdot \underline{p(c)} \cdot \underline{p(d)} = p(cdf)$
ϕ_{c_4}	$\underline{p(de)}, \underline{p(f\|d)}$	$\underline{p(de)} \cdot \underline{p(f\|d)} = p(def)$
ϕ_{c_5}	$\underline{p(f)}$	$p(h\|f) \cdot \underline{p(f)} = p(fh)$
ϕ_{c_6}	$\underline{p(ef)}$	$p(g\|ef) \cdot \underline{p(ef)} = p(efg)$

Table 1: Allocating separator marginals, the underlined terms are either the separator marginals or from the factorization of a separator marginal.

the clique potential $\phi_{c_6}(efg) = p(g|ef)$, we can multiply it with the separator marginal $p(ef)$ which results in $\phi_{c_6}(efg) = p(g|ef) \cdot p(ef) = p(efg)$. So far, we have successfully obtained marginals for cliques c_1, c_2, c_5, and c_6 and we have consumed the separator marginals $p(f)$ and $p(ef)$ during this process. We still need to make the remaining clique potentials $\phi_{c_3}(cdf) = p(f|cd)$ and $\phi_{c_4}(def) = 1$ marginals by consuming the remaining separator marginals, i.e., $p(c)$, $p(de)$ and $p(df)$. In order to make $\phi_{c_3}(cdf) = p(f|cd)$ marginal, we need to multiply it with $p(cd)$, however, we only have the separator marginals $p(c)$, $p(de)$ and $p(df)$ at our disposal. It is easy to verify that we cannot mingle $p(de)$ with $\phi_{c_3}(cdf) = p(f|cd)$ to obtain marginal $p(cdf)$. Therefore, $p(de)$ has to be allocated to the clique potential $\phi_{c_4}(def)$ such that $\phi_{c_4}(def) = 1 \cdot p(de)$. We now only have $p(df)$ at our disposal for making $\phi_{c_4}(def)$ a marginal. Note that $p(df) = p(d) \cdot p(f|d)$, and this factorization of the separator marginal helps make $\phi_{c_4}(def) = p(de)$ a marginal by multiplying $p(f|d)$ with $\phi_{c_4}(def)$ to obtain $\phi_{c_4}(def) = p(de) \cdot p(f|d)$. It is perhaps worth pointing out that the CI $I(f,\ d,\ e)$ holds in the original DAG in Fig. 1. Therefore, $\phi_{c_4}(def) = p(def)$. We are now left with the separator marginal $p(c)$ and $p(d)$ (from the factorization of the separator marginal $p(df)$) and the clique potential $\phi_{c_3}(cdf)$, and $p(c)$ and $p(d)$ have to be multiplied with $\phi_{c_3}(cdf)$ to yield $\phi_{c_3}(cdf) = p(f|cd) \cdot p(c) \cdot p(d)$. Again, since CI $I(d,\ \emptyset,\ c)$ holds in the original DAG in Fig. 1, $\phi_{c_3}(cdf) = p(cdf)$. We have thus so far successfully and algebraically used all separate marginals to transform each clique potential ϕ_{c_i} into a marginal $p(c_i)$. Table 1 summarizes the allocation scheme for the multiplied separator marginals.

3.2 Observations

One may perhaps consider the success of the example in Sect 3.1 as a sheer luck. In the following, we will show that this is not a coincidence.

According to the GP method in the Hugin architecture, every clique c_i in the junction tree was initially associated with a clique potential ϕ_{c_i}. During the course of propagation, the clique receives messages from all its neighbors, and the clique potential ϕ_{c_i} multiplies with all these received messages. The result of the multiplication is $p(c_i)$. In other words, the GP method transforms the clique potential ϕ_{c_i} into a clique marginal $p(c_i)$. This algorithmic phenomena of the GP method can be explained algebraically. Consider Eq. (7), in which the numerators are the original clique potentials together with the multiplied separator marginals. Every clique potential has to mingle with some appropriate separator marginal(s) or its factorization if necessary to be transformed into a marginal. The messages received by all the cliques in the GP method as a whole, which *algorithmically* transform each clique potential into a clique marginal, have the same effect as the separator marginals we multiplied in Eq. (7), which *algebraically* transform each clique potential into a clique marginal. This analysis leads to the following proposition.

Proposition 1 *The product of the messages received by every clique in the GP method equals to the product of all separator marginals.*

Recall the motivating example in Sect 3.1, in which we successfully mingle the separator marginals with the clique potential. This perfect arrangement of separator marginals is not a coincidence, in fact, it can always be achieved as we explain below.

Assigning either a separator marginal or its factorization to a clique potential, as shown in Sect 3.1, must satisfy one necessary condition, namely, condition (1) of Definition 1, in order for the product of the clique potential with the allocated separator marginal or its factorization to be a marginal. That's to say, for each ϕ_{c_i}, we need a CPD with a_j as head for each $a_j \in c_i$. If a variable, say a_j, appears m times in m cliques in the junction tree, then each of these m cliques will need a CPD with a_j as head. However, the original BN only provides one CPD with a_j as head, and we are short of $m-1$ CPDs (with a_j as head). Fortunately, m cliques containing a_j implies the junction tree must have exactly $m-1$ separators containing the variable a_j (Huang & Darwiche 1996), therefore the $m-1$ needed CPDs with a_j as head will be supplied by the $m-1$ separator marginals (or their factorizations). This analysis leads to a simple procedure to allocate separator marginals.

Procedure: Allocate Separator Marginals (ASM)

Step 1. Suppose the CPD $p(a_i|\pi_{a_i})$ is assigned to a clique c_k to form ϕ_{c_k}. If the variable a_i appears in a separator s_{kj} between c_k and c_j, then draw a small arrow originating from a_i in the separator s_{kj} and pointing to the clique c_j. If variable a_i also appears in other separators in the junction tree, draw a small arrow on a_i in those separators and point to the neighboring clique away from clique c_k's direction. Repeat this for each CPD $p(a_i|\pi_{a_i})$ of the given BN.

Step 2. Examine each separator s_i in the junction tree, if the variables in s_i all pointing to one neighboring clique, then the separator marginal $p(s_i)$ will be allocated to that neighboring clique , otherwise, $p(s_i)$ has to be factorized so that the factors in the factorization can be assigned to appropriate clique indicated by the arrows in the separator.

The procedure ASM can be illustrated using Fig. 3. If all variables in the same separator are pointing to the same neighboring clique, that means the separator marginal as a whole (without being factorized) will be allocated to the neighboring clique, for example, the separator marginal $p(c)$, $p(de)$, $p(f)$, and $p(ef)$ in the figure. If the variables in the separator are pointing to different neighboring cliques, that means the separator marginal has to be factorized before the factors in the factorization can be allocated according to the arrow. For example, the separator marginal $p(df)$ has to be factorized so that the factor $p(d)$ is allocated to $\phi_{c_3}(cdf)$ and $p(f|d)$ is allocated to $\phi_{c_4}(def)$. (The factorization of a separator marginal will be further discussed shortly.)

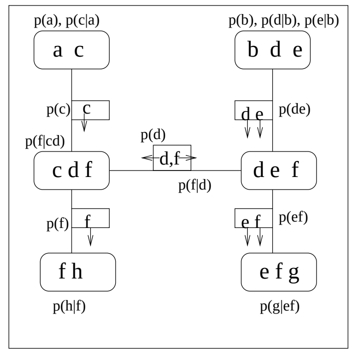

Figure 3: Allocating separate marginals by ASM.

Proposition 2 *For each separator in a junction tree, one can always assign either the separator marginal or some factors in its factorization to an appropriate clique c_i as dictated by the procedure ASM, such that for each variable $a_j \in c_i$, there is a CPD assigned/allocated to the clique ϕ_{c_i} in which a_j is the head.*

Although an appropriate allocation of the separator marginals can always be guaranteed to satisfy condition (1) of Definition 1, one still needs to show that such an allocation will not produce a directed cycle when verifying condition (2) of Definition 1. It is important to note that a directed cycle can be created in a directed graph if and only if one draws a directed edge from the descendant of a node to the node itself.

Consider a clique c_i in a junction tree and its neighboring cliques. Between c_i and each of its neighboring clique, say clique c_j, is a separator s_{ij} whose separator marginal $p(s_{ij})$ or some factors in its factorization can possible be allocated to the clique potential ϕ_{c_i}. As the example in Sect 3.1 shows, sometimes, the separator marginal ϕ_{c_i} as a whole will be allocated to ϕ_{c_i}; sometimes, some factors in the factorization of $p(s_{ij})$ will be allocated to ϕ_{c_i}. Suppose the separator marginal $p(s_{ij})$ is allocated to ϕ_{c_i}. If one follows the rule of condition (2) in Definition 1 to draw directed edges based on the original CPDs assigned to ϕ_{c_i} and the newly allocated separator marginal $p(s_{ij})$, no directed cycle will be created, because the original CPDs assigned to ϕ_{c_i} are from the given BN, which will not cause any cycle, and the variables in s_{ij} will be ancestors of all other variables in the clique, which will not create any cycle as well. Suppose the separator marginal $p(s_{ij})$ has to be factorized first as a product of CPDs, and only some of the CPDs in the factorization will be allocated to c_i (and the rest will be allocated to c_j). In this case, it is very possible that the CPDs in the factorization allocated to c_i will cause a directed cycle. For example, in the example in Sect 3.1, we decomposed the separator marginal $p(df)$ as $p(df) = p(d) \cdot p(f|d)$. In fact, we could have decomposed it as $p(df) = p(f) \cdot p(d|f)$ and assigned the factor $p(d|f)$ to c_3, which would result in $\phi_{c_3}(cdf) = p(c) \cdot p(f|d) \cdot p(f|cd)$. It is easy to verify that ϕ_{c_3}, after incorporating the allocated CPD $p(d|f)$, satisfies the condition (1) but not (2) of Definition 1, which means that $\phi_{c_3}(cdf) = p(c) \cdot p(f|d) \cdot p(f|cd) \neq p(cdf)$ and it is not a Bayesian factorization. It is important to note that the factorization $p(df) = p(f) \cdot p(d|f)$ does not follow the topological ordering of the variables d and f (d should precede f in the ordering) with respect to the original DAG, in which f is a descendant of d. Drawing a directed edge from f to d, as dictated by the CPD $p(d|f)$, would mean a directed edge from the descendant of d, namely, the variable f to the variable d itself, and this is exactly the cause of creating a directed cycle. However, if we factorize $p(df)$ as we did in Sect 3.1, there will be no problem. This is because when we factorize $p(df)$ as $p(df) = p(d) \cdot p(f|d)$, we were following the topological ordering of the variables d and f with respect to the original DAG such that the heads of the CPDs in the factorization are not ancestors of their respective tails in the original DAG. This analysis leads to the following proposition, which is a continuation of the previous proposition.

Proposition 3 *If the procedure ASM indicates that a separator marginal $p(s_i)$ has to be factorized before it can be allocated to its neighboring cliques, then $p(s_i)$ must be factorized based on a topological ordering of the variables in s_i with respect to the original DAG.*

3.3 Demystify the Messages

In Proposition 1, we have established a rough connection between the messages passed in the GP method and the separator marginals. We point out that the product of all the messages is equal to the product of all separator marginals. Propositions 2 and 3 further explored this rough connection. Jointly, Propositions 2 and 3 suggest that all the separator marginals or their factorizations can be appropriately allocated to clique potentials, so that each clique potential, multiplying with the allocated, results also in the desired clique marginal. That is to say, the messages received by each clique algorithmically in the GP method are equal to the allocated separator marginal or its factors received by each clique potential algebraically. In the following, we will present the main contribution of this paper. We will show exactly what a message really is in the GP method.

Let c_i and c_j be two cliques in a junction tree and s_{ij} be the separator between c_i and c_j. Regardless of which clique in the junction tree is chosen as the root, there are two messages that will be passed between c_i and c_j. Without loss of generality, suppose a message denoted $m_{i\rightarrow j}$ is passed from c_i to c_j in the Collect-Evidence stage, and another message denoted $m_{i\leftarrow j}$ is passed in the Distribute-Evidence stage.

Theorem 1 [2] *Consider the result of applying the procedure ASM to the junction tree. There are three possible outcomes regarding the separator marginal $p(s_{ij})$.*

(a) If $p(s_{ij})$ as a whole is allocated to c_j, then $m_{i\rightarrow j} = p(s_{ij})$ and $m_{i\leftarrow j} = 1$.

(b) If $p(s_{ij})$ as a whole is allocated to c_i, then $m_{i\rightarrow j} = 1$ and $m_{i\leftarrow j} = p(s_{ij})$.

(c) If $p(s_{ij})$ has to be factorized (following a topological ordering of variables in s_{ij}), then $m_{i\rightarrow j}$ = the product of factors allocated to c_j and $m_{i\leftarrow j}$ = the product of factors allocated to c_i.

We use an example to illustrate the theorem.

Example 2 Consider the junction tree in Fig. 2 (i). If clique $c_4 = def$ is chosen as the root for the GP method, then $c_3 = cdf$ will send a message to c_4 during the Collect-Evidence stage and c_4 will send a message to c_3 during the Distribute-Evidence stage. Before c_3 can send the message to c_4, cliques $c_1 = ac$ and $c_5 = fh$ have to pass messages to c_3. The message from c_1 to c_3 is $\phi_c = (\sum_a p(a) \cdot p(c|a))/1 = p(c)$, which coincides with (a) in the above theorem. The message from c_5 to c_3 is $\phi_f = (\sum_h p(h|f))/1 = 1$, which coincides with (b) in the above theorem . The clique c_3, after absorbing these two messages, becomes $\phi_{c_3} = p(f|cd) \cdot p(c) \cdot 1 = p(f|cd) \cdot p(c)$. The message sent from c_3 to c_4 is $\phi_{df} = (\sum_c p(f|cd) \cdot p(c))/1 = \sum_c \frac{p(fcd)}{p(cd)} \cdot p(c) = \sum_c \frac{p(fcd)}{p(c) \cdot p(d)} \cdot p(c) = \sum_c \frac{p(fcd)}{p(d)} = p(fd)/p(d) = p(f|d)$, which coincides with (c) in the above theorem.

4. Passing Much Less Messages for Inference

The revelation of the messages in the GP method suggests a new approach to compute the clique marginals. The idea comes from the example in Sect 3.1 and Table 1, in which it demonstrated that one only needs to multiply the originally assigned CPDs of a clique with the allocated separator marginal(s) or the factors in its (their) factorization(s) suggested by the procedure ASM, in order to obtain the clique marginal. Although the originally assigned CPDs, namely, those in Eq. (3), are always available from the given BN, the allocated separator marginal(s) or its(their) factors are *not*. However, should they become available, calculating the marginal for a clique then becomes the simple task of multiplication as shown in Table 1.

Consider Fig. 3, it is noted that for every clique in the junction tree, either it needs to send the separator marginal or the factors in its factorization to its neighboring cliques once the clique marginal is known (for example clique $c_1 = ac$ needs to send $p(c)$ to clique $c_3 = cdf$ if $p(ac)$ is known) , or it needs to receive the allocated separator marginal or the factors in its factorization from its neighboring cliques (for example clique $c_3 = cdf$ needs to receive $p(c)$ and $p(f|d)$ from cliques c_1 and $c_4 = def$, respectively), in order to transform the clique potential into the clique marginal.

It is further noted that some clique potentials are clique marginals automatically without needing to receive anything from its neighboring separators. For example, the clique potentials for $c_1 = ac$ and $c_2 = bde$ in Eq. (3) are already marginals, as shown in the first two rows in Table 1. Once $p(ac)$

[2] Due to limited space, the proof of the theorem will appear in an extended version of this paper.

and $p(bde)$ are available, they can now send the needed separator marginals $p(c)$ and $p(de)$ to the clique potentials c_3 and c_4, respectively. At this point, clique potentials c_3 and c_4 further need the factors in the factorization of the separator marginal $p(df)$ from each other. Clique c_3 needs the factor $p(d)$ to transform ϕ_{c_3} into $p(c_3)$, and clique c_4 needs $p(f|d)$ to transform ϕ_{c_4} into $p(c_4)$. If $p(c_4)$ is known, then $p(d)$ can be supplied to c_3; if $p(c_3)$ is known, then $p(f|d)$ can be supplied to c_4. Unfortunately, both $p(c_3)$ and $p(c_4)$ are unknown at this point. This seems to be a deadlock situation. Ideally, if clique c_3 can somehow receive the needed factor $p(d)$ not from the unknown $p(c_4)$ but from the known ϕ_{c_4} and clique c_4 can somehow receive the needed factor $p(f|d)$ not from the unknown $p(c_3)$ but from the known ϕ_{c_3}, then $p(c_3)$ and $p(c_4)$ can both be computed. [3] Once $p(c_3)$ and $p(c_4)$ are available, they can then send the separator marginals $p(f)$ and $p(ef)$ to cliques c_5 and c_6 respectively. Receiving the needed separator marginals $p(f)$ and $p(ef)$, ϕ_{c_5} and ϕ_{c_6} become $p(c_5)$ and $p(c_6)$ as shown in the 5th and 6th rows in Table 1.

From the above analysis, obviously, each clique potential becomes clique marginal once the clique receives all its needed from its neighboring separators. If we consider the allocated marginal or the factors in its factorization received by a clique from its neighboring clique as a message, then it is easy to verify that there is no need to pass $2(n-1)$ messages as in the GP method (recall that n denotes the number of cliques in a junction tree and $n = 6$ in the example in Sect 3.1). In fact, applying the GP method on the example in Sect 3.1 requires passing $(6-1) \times 2$=10 messages; our analysis above shows that only 6 messages are really needed. The other four messages passed by the GP method are identity function 1 according to Theorem 1, which has no effect on the receiving cliques. The revealed semantic meaning of the messages helps save a significant amount of computation required by the GP method.

We have conducted a preliminary experiment on a number of publicly available BNs. The experimental data is in Table 2. It can be seen that by utilizing the semantic meaning of the messages, we can save up to 50% of messages that would have had to be passed by the GP method. This suggests that propagation based on allocating separator marginals could be more efficient than the GP method.

5. Conclusion

In this paper, we have studied the messages passed in the GP method algebraically. It was revealed that the messages are actually separator marginals or factors in their factorizations. Passing messages in the GP method can be equivalently considered as the problem of allocating separator marginals. This different perspective of propagation gives rise to a different idea of computing clique marginals. Our preliminary experimental results are very encouraging. In all the BNs tested, much less number of messages need to be passed compared with the GP method. Since passing messages is the basic operation in the propagation algorithm for computing clique marginals, our experimental results seem to suggest that a more efficient method for inference can possibly be designed based on the semantic meaning of the messages revealed in this paper.

[3] We have developed such technique to obtain the needed factors in the factorization of the separator marginal when the deadlock situation occurs. Due to limited space, it will be reported in the extended version of the paper.

Network			**Total Messages**		**% of**
	nodes	cliques	Hugin Method	Our Method	**Savings**
Asia	8	6	10	6	40%
Car_ts	12	6	10	5	50%
Alarm	37	27	52	33	37%
Printer_ts	29	11	20	10	50%
Mildew	35	29	56	47	16%
4sp	58	40	78	58	26%
6hj	58	41	80	57	29%
r_choice	59	42	82	57	30%
Barley	48	36	70	59	16%
Munin2	1003	868	1734	1190	31%
Munin3	1044	904	1806	1220	32%
Munin4	1041	876	1750	1163	34%

Table 2: Comparison of message counts on various networks

Acknowledgment

The authors wish to thank referees for constructive criticism and financial support from NSERC, Canada.

References

Huang, C., and Darwiche, A. 1996. Inference in belief networks: A procedural guide. *International Journal of Approximate Reasoning* 15(3):225–263.

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

Jensen, F. 1996. *An Introduction to Bayesian Networks*. UCL Press.

Lauritzen, S., and Spiegelhalter, D. 1988. Local computation with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society* 50:157–244.

Lepar, V., and Shenoy, P. P. 1998. A comparison of Lauritzen-Spiegelhalter, Hugin, and Shenoy-Shafer architectures for computing marginals of probability distributions. In Cooper, G. F., and Moral, S., eds., *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 328–337. San Francisco: Morgan Kaufmann.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, California: Morgan Kaufmann Publishers.

Shafer, G. 1991. An axiomatic study of computation in hypertrees. School of Business Working Papers 232, University of Kansas.