Optimizing Local Computation for Cooperative Probabilistic Reasoning

Karen H. Jin
Faculty of Computer Science
Dalhousie University
Halifax, NS
Canada B3H 3J5

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

Abstract

Multiply Sectioned Bayesian Networks (MSBNs) extend single-agent Bayesian networks to the setting of multi-agent probabilistic reasoning. The MSBN global propagation is conducted through inter-agent message passing, coupled with intra-agent (local) message passing at local domains. Existing LJF-based MSBN inference algorithms require repeated full-scale local propagation, which may cause bottlenecks in a non-sparse network. We propose a novel method that conducts 1) delayed inter-agent message manipulation, and 2) partial local message propagation. Analysis shows that our approach significantly reduces the amount of local computation while maintaining the correctness of MSBN global propagation.

1 Introduction

As an extension to the traditional Bayesian Network (BN), a Multiply Sectioned Bayesian Network (MSBN) provides a specific framework for probabilistic reasoning in a multi-agent setting (Xiang 2002). A large and distributed problem can be modeled as a set of organized subdomains each maintained by an individual light-weighted agent. An MSBN's hypertree structure enables the extension from the BN Junction Tree (JT) algorithm to agent-based inference. Typically, a Linked Junction Forest (LJF) is constructed for distributed inference in an MSBN. Inter-agent messages, passed between two LJF local JTs over their interface, are essential in unifying the beliefs of different MSBN subnets. Each inter-agent message contains a set of potentials over an LJF linkage tree representation of the shared nodes.

Existing MSBN LJF inference algorithms mostly follow the same underlying principle of the product-based Hugin architecture. In particular, the sender of an inter-agent message must conduct a full round of local JT propagation to ensure the correctness of the message; the receiver of the message must immediately propagate the message through a round of full-scale local propagation as well. Methods have been proposed to improve local computation, e.g. adopting lazy inference in message calculation (Xiang, Jensen, and Chen 2006). However, repeated local propagation is still required. This results in extensive local message calculation which may form communication bottlenecks and possibly halt the MSBN global inference in a dense network.

In this paper, we introduce two techniques to optimize the LJF local computation.

- **Delayed message manipulation** eliminates repeated local propagation. Despite the similarities between a BN JT and an MSBN hypertree, there has been little research on extending non-Hugin message passing methods to the MSBN LJF global inference. In this paper, we introduce new semantics for the LJF linkage trees. A pair of corresponding linkage trees, used as a single Hugin separator in existing architectures, are now treated as two individual buffers. Then we present a new architecture, extending from the Shenoy-Shafer architecture to avoid repeated rounds of local message passing.
- **Partial local propagation** further reduces the amount of local messages passed. During the MSBN global propagation, we achieve the globally coherent system belief for each agent through inter-agent message passing. Existing algorithms must repeatedly invoke local propagation to maintain a state of consistency, which in term guarantees the proper handling of inter-agent messages. In this paper, however, we present algorithms that allow us to obtain correct inter-agent messages and consistent local beliefs without *any* full-scale local propagation. The technique of partial propagation, previously studied in more restricted context of updating LJF linkage potentials (Xiang 1995) and LJF initial calibration (Jin and Wu 2008), is now applied on a complete local JT and during the actual global belief propagation.

2 Background

2.1 MSBNs

We assume the readers are familiar with common terminologies presented in the literature of BN. MSBNs extend BNs and provide a framework for uncertainty reasoning in cooperative multi-agent systems (Xiang 2002). An MSBN is composed of a set of BN subnets each maintained by an agent and representing a partial view of a larger problem domain. The union of all subnet DAGs must also be a DAG, and these subnets are organized into a tree structure called a hypertree. Each hypertree node corresponds to a subnet, and each hypertree link corresponds to a *d-sepset*, which is the

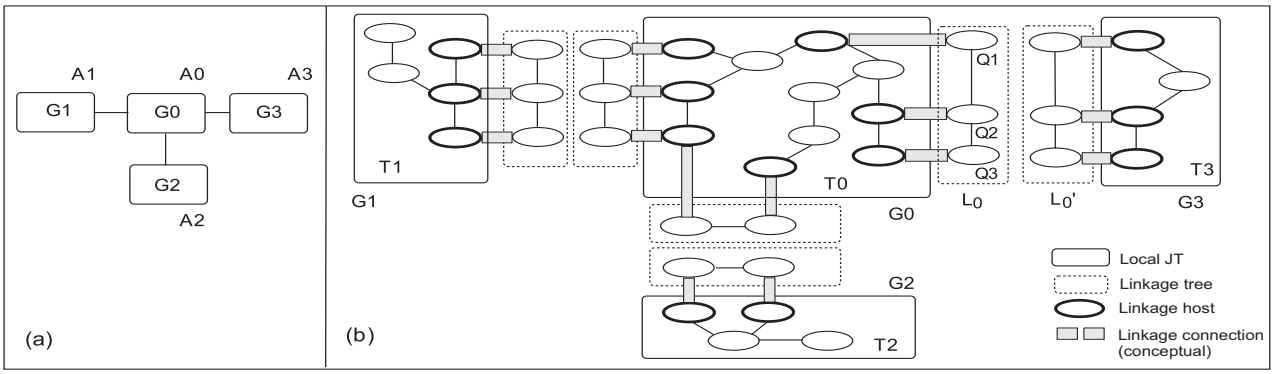

Figure 1: MSBN Inference with LJFs. (a) The hypertree; (b) The constructed LJF with linkage trees.

set of shared variables between adjacent subnets. A hypertree follows the running intersection property so that a hyperlink renders two sides of the network conditionally independent similar to a JT separator. Several meta-requirements are necessary for the MSBN representation and they distinguish MSBNs from several other knowledge representation models (Xiang 2002).

A secondary structure called *linked junction forest* (LJF) is typically used for distributed inference in MSBNs. An LJF is constructed through a process of cooperative and distributed compilation, so that each node in a hypertree is transformed into a local JT and each hyperlink into a *linkage tree*. A linkage tree is also a JT, in which each cluster is called a *linkage*, and each separator, a *linkage separator*. The cluster in the local JT that contains a linkage is called a *linkage host*. Essentially, a linkage tree is an alternative representation of the d-sepset between adjacent subnets, but contains linkages with a smaller domain than the d-sepset in order to facilitate inter-agent message calculation. Each of the adjacent LJF subnet pair maintains its own linkage tree corresponding to the same d-sepset. These two linkage trees may be different, but only at their topologies so the result of message passing is not affected (Xiang 2002).

For example, a trivial MSBN with 4 subnets is shown in Figure 1 (a). Three hyperlinks connect the subnets into a hypertree structure. We have omitted the details of each BN subnet structure for simplicity. Figure 1 (b) shows an LJF constructed from the MSBN in Figure 1 (a). Local JTs, $T_i, i = 0, ..., 3$ are enclosed by boxes with solid edges. Linkage trees, converted from d-sepsets, are enclosed by boxes with dotted edges. The figure also shows the linkages of each linkage tree and their corresponding linkage hosts in the local JT.

2.2 Local Computation in Hugin-based Distributed Inference

The main task of MSBN inference is to supply the correct global posterior probabilistic knowledge to each agent given all locally observed evidence. The LJF-based algorithms, extending from JT-based Hugin architecture (Jensen, Lauritzen, and Olesen 1990), have been proven to be the most successful (Xiang 2002).

Existing LJF inference algorithms (Xiang 2002) all follow the Hugin architecture for inter-agent message passing. Beliefs are propagated as inter-agent messages calculated over the linkage trees connecting two adjacent subnets. For an LJF local JT T_i to deliver a message to T_j over T_i's linkage tree L_{ij}, each linkage Q_i in L_{ij} is assigned an *extended linkage potential* and together all the potentials compose an inter-agent message. For example, consider the LJF with local JTs and linkage trees shown in Figure 1 (b). An inter-agent message, originated from G_0 to be delivered to G_3, is calculated over their corresponding linkage trees L_0 and L_0'. The message consists of the extended potentials over three linkages Q_1, Q_2 and Q_3.

The local belief of each subnet must be updated relative to the global system belief by absorbing incoming messages from its adjacent subnets. The Hugin-based global inference consists of a coordinated sequence of inter-agent message passing for all subnets. Typically, a random root agent is selected, and two rounds of inter-agent messages passing are recursively carried out amongst all agents. However, each inter-agent "Hugin" message is no longer passed between two JT clusters as in its original context. Now, we have two subnets each having its own internal structure of a local JT. Therefore, the inter-agent message passing must be coupled with local message passing. With the existing Hugin-based architectures, a round of local updates is required to bring subnets to local consistency for the purpose of calculating an outgoing inter-agent message, as well as absorbing an incoming one. This obviously results in a significant amount of local message passing in local JTs.

We show such extensive local message passing with an example. In Figure 1 (b), suppose G_0 is the root. During inward message passing, G_0 will receive three inter-agent messages, one from each of its three neighbors, G_1, G_2 and G_3. First, these three subnets must achieve their local consistency to obtain the inter-agent messages for G_0. More importantly, G_0 must update its local belief upon the arrival of *each* inter-agent message, thus for a total of three times. Each local propagation involves a full round of local message passing, but the resulting local JT is not consistent with system belief until after the very last update.

We may view the multiple local propagation as a means

to establish a transitional state of local consistency within a local JT. Given such transitional consistency, however, the globally consistent posterior belief is not available until adequate amount of incoming messages have been received from neighboring agents. Nevertheless, we can not forgo the repeated local updates due to the fact that each pair of linkage trees are used as a single Hugin separator. An inter-agent message coming from one direction must be immediately processed (i.e. locally propagated) since its linkage tree storage will be soon occupied by another message from the opposite direction. The Hugin-based architectures must maintain the costly transitional consistency in order to handle the inter-agent message passing correctly.

3 A New LJF Inference Architecture

Aiming at improving the efficiency of local computation, we present a new LJF global inference architecture. Our architecture is based on the same underlying LJF construct, but extends the Shenoy-Shafer algorithm (Shenoy and Shafer 1986). Local propagation is optimized in terms of both the total number of propagations and the required amount of local message passing.

3.1 Linkage Tree as Message Buffer

Given two adjacent agents in an LJF, each agent maintains its own linkage tree. These two linkage trees are constructed over the same d-sepset, thus containing a same set of linkages and linkage separators. During the global propagation with existing algorithms, the two corresponding linkage trees serve the purpose of a *single* Hugin separator. This treatment guarantees the correctness of the Hugin-based message passing. However, as an inter-agent message is delivered between two subnets each with an internal JT structure, maintaining a single conceptual separator demands immediate process of the message. That is, a complete round of local message passing must be conducted repeatedly.

Contrary to the Hugin architecture, messages in Shenoy-Shafer architecture are explicitly calculated and stored. That is, two message buffers are allocated for each pair of adjacent JT clusters. Extending the idea to the LJF structure, we can simply view each linkage tree as a message buffer. Indeed, it is natural to treat LJF linkage trees as message buffers as each of them is maintained independently by an agent and no additional storage is required.

Inter-agent message passing under our new architecture is now buffered into the linkage tree. Let A_i and A_j be two adjacent agents. An inter-agent message M passed from A_i to A_j is first obtained from A_i's linkage tree L_i. M consists of a set of potentials for all linkages in L_i. Next, M is delivered to A_j and stored at A_j's corresponding linkage tree L_j. That is, each potential of M is stored in the corresponding linkage of L_j. Since the local propagation is no longer triggered by any incoming message, an agent determines when to perform the local message passing following certain global propagation rules.

We use a standard non-rooted message scheduling scheme (Shenoy and Shafer 1986). Rather than following a recursive call during the global propagation, each agent starts to process incoming and outgoing inter-agent messages simultaneously. The coordination of the message passing is controlled by two simple rules:

-*Rule 1.* When an agent has received all except one message from its adjacent agents, the agent calculates an outgoing message to that particular neighbor.

-*Rule 2.* When an agent has received the last message from its adjacent agents, the agent absorbs the message, updates the local belief and calculates all outgoing messages to its other neighbors.

Based on the above two rules, we present Algorithm $Communicate\ Belief$ that runs at each individual agent during the global propagation process.

Algorithm 1 $Communicate\ Belief$

Let $T_i (i = 1, ..., n)$ be the local JTs and $\mathcal{H}$ the corresponding hypertree of an LJF L, which is populated by n agents with one at each subnet. Each agent A_i has K_i adjacent subnets. When called, each agent A_i performs the following:

1. *Set cnt= K_i;*
2. *While($cnt \neq 0$) {*
3. *Wait for incoming messages;*
4. *If an incoming message is received*
5. *Set $cnt = cnt - 1$;*
6. *If ($cnt == 1$ && $\exists$ A_j such that incoming*
7. *message buffer for A_j is empty)*
8. *Deliver the first outgoing message to A_j;*
9. *}*
10. *Conduct local propagation in T_i;*
11. *Deliver all remaining outgoing messages;*

The global belief propagation with Algorithm $Communicate\ Belief$ is controlled by an implicit order of inter-agent message calculation. Repeated local propagation is avoided. Regardless of the network topology, each agent requires only two rounds of local propagations: one at Line 10 after all messages have arrived, and one at Line 8 when local propagation is implicitly needed to calculate the message. We show in the next sections that these two rounds of local message passing can be further optimized with partial local propagation.

3.2 Partial Local Propagation

An agent's correct local belief, w.r.t the LJF global system domain, is obtained through receiving inter-agent messages from its adjacent agents. Under the Hugin-based architectures, local consistency is required to 1) compose an outgoing message and 2) absorb an incoming message. The local consistency is costly maintained: a full round of local JT propagation. Nevertheless, a consistent local JT *during* the global propagation process does not reflect the global consistency, until all incoming messages have arrived. Chain-pattern partial propagation was proposed to facilitate the absorbance of an incoming message over the linkage tree (Xiang 1995), but it is restricted to the context of host trees and has limited impact on reducing the total cost of local propagation under the Hugin architecture.

In our new architecture, it is not necessary to constantly maintain such transitional consistency. A local JT is kept partially consistent for most of the time during the global propagation. We first introduce the concept of a *linkage host tree* and an *extended linkage host tree* to support the partial propagation.

Definition 1 Linkage host tree. *Let T_i be a local JT and L_i be a linkage tree connecting T_i to an adjacent local JT. In T_i, H is the set of linkage host clusters for linkages in L_i. Then a JT T_h can be constructed by connecting all clusters of H, and T_h is called a linkage host tree of T_i w.r.t. L_i.*

For example, consider again the subnet G_0 in Figure 1 (b). Figure 2 shows the local JT T_0 and the linkage tree L_0 to its neighbor T_3. Suppose the root cluster of T_0 is C_r. The linkage hosts for linkage tree L_0 are clusters C_r, C_1 and C_3, each shown with a bold border. These three clusters can be linked into a JT in which the dashed line indicates the extra link needed. This JT is the linkage host tree of T_0 w.r.t. L_0.

Definition 2 Extended linkage host tree. *Let T_i be a local JT and its linkage host tree T_h that corresponds to a linkage tree L_i. The minimum subtree of T_i that contains all clusters in T_h is called an extended linkage host tree T_e of T_i with regards to L_i.*

In Figure 2, linkage host clusters C_r, C_1 and C_3, together with a non-host cluster C_2, construct an extended linkage host tree T_e. T_e is shown in the shaded area of T_0.

Note that a linkage host tree is a conceptually defined structure that contains *only* linkage host clusters. It may not be a subtree of the local JT. Meanwhile, an extended linkage host tree is a subtree within the local JT, containing clusters possibly other than linkage hosts.

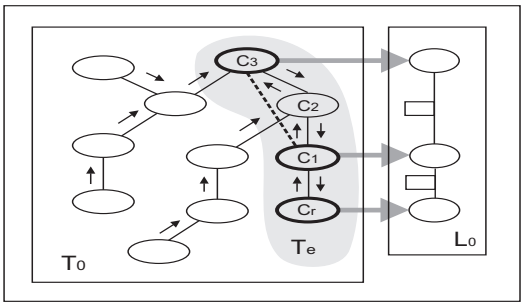

Figure 2: The local JT T_0 from Figure 1, shown with linkage tree L_0, linkage host tree L_h and extended linkage host tree T_e. The figure also shows an example of the partial local propagation for calculating the first outgoing message.

Partial Propagation for Message Calculation

An outgoing inter-agent message contains a set of linkage potentials each obtained from its corresponding linkage host. During message calculation, all linkage hosts, in the linkage tree w.r.t. to the receiver of the message, should contain the correct potential. Rather than maintaining a consistent local JT, we only need to make sure that all linkage hosts are updated with current beliefs. Thus, we optimize the original full-scale local propagation to a partially conducted local message passing as described in Algorithm Cal_Single_MSG

Algorithm 2 Cal_Single_MSG**.**

Let A_i and A_j be two adjacent agents. A_i's local JT is T_i with the set of clusters C. T_i's linkage tree to A_j is L. In T_i, H is the set of linkage hosts of L and the extended linkage host tree is T_e. When called, an inter-agent message passed from A_i to A_j is calculated as the following:

Step 1. For all linkage trees of A_i connecting to neighboring agents except A_j, absorb the extended linkage potential and update the belief on corresponding linkage hosts.

Step 2. Randomly select a linkage host $C_r \in H \subset C$ as the root of T_e as well as T_i. Direct all clusters away from C_r in T_i.

Step 3. Perform an inward message passing on C_r in T_i, such that C_r calls recursively all child clusters to send an inward message.

Step 4. Perform a restricted outward message passing on C_r within the context of T_e, such that C_r sends outward messages to all linkage hosts recursively.

Step 5. For each linkage in L, obtain corresponding extended linkage potential and compose the outgoing message.

Consider again Figure 2. Suppose we want to calculate an outgoing message from T_0 over linkage tree L_0. After Step 1 of Algorithm Cal_Single_MSG, we select C_r and perform an inward message passing toward C_r in the local JT T_0. The arrows between the clusters show the message flow, which is originated from all leave nodes and with messages recursively passed toward the root. Next, outward message passing is originated from root C_r, but the outward messages only reach the clusters within the extended linkage host tree T_e. After this phase of partial message passing, we can compose an outgoing message over L_0 through the linkage host of each linkage in L_0, as shown with the shaded thick arrows.

We may select any root for the partial local propagation, as long as the outward passing covers the extended linkage tree. However, it is most efficient to root the partial message passing at a linkage host. Given such a root, we conduct a full inward passing so the local messages carrying the probability information of all JT clusters flow towards the root. Then a partial outward passing is performed only in the context of the extended linkage host tree to distribute belief originated from the root within the clusters of the extended linkage host tree. The local JT is not consistent as the propagation is not complete. Nevertheless, the extended linkage host tree is consistent, so that all linkage hosts are equipped with the correct local knowledge to form the outgoing inter-agent message.

Partial Propagation for Local Consistency

When an agent receives all incoming messages, the local belief with all messages absorbed is consistent to the global system belief. As mentioned earlier, the local belief update

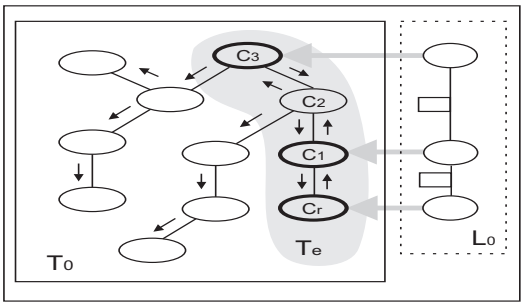

Figure 3: An example of partial local propagation for updating local belief upon the arrival of last incoming message. Shown with the local JT T_0, linkage tree L_0 and extended linkage host tree T_e.

at Line 10 of Algorithm $Communicate\_Belief$ can also be replaced as a partial propagation. The partial local propagation was used for sending a single outgoing message, and we extend it to incorporate all incoming messages.

In fact, a closer look at Algorithm $Communicate\_Belief$ reveals that we need to deal with only one additional incoming message after a call to Algorithm Cal_Single_MSG. Moreover, the sender of this message is also the exact receiver of the outgoing message just calculated by the call to Cal_Single_MSG. Recall that by the time a call to Algorithm Cal_Single_MSG is finished, all previously received messages have already been absorbed into the extended linkage host tree. Thus, we only need to update the *same* extended linkage host tree to absorb this last message.

Essentially, we have taken full advantage of the non-rooted message scheduling scheme (*Rule* 1 and *Rule* 2) in our new architecture. Algorithm Cal_Single_MSG consists of full inward (in the context of local JT) and partial outward (in the context of the extended linkage host tree) message passing. Similarly, we design another partial local propagation algorithm in order to achieve local consistency upon receiving the very last message.

Algorithm 3 $Update\_Belief$**.**

Let A_i and A_j be two adjacent agents. A_i's local JT is T_i with the set of clusters C. T_i maintains a linkage tree L to A_j. In T_i, H is the set of linkage hosts of L and the extended linkage host tree is T_e. Suppose A_i has calculated an outgoing message to A_j with Algorithm Cal_Single_MSG with selected root C_r. A_i, on receiving the last incoming message from A_j, performs the following:

Step 1. A_i absorbs message from A_j by updating belief of all clusters in H with the extended linkage potential of each linkage in L.

Step 2. Perform a restricted inward message passing on C_r in T_e, such that C_r calls recursively all clusters in H to send an inward message.

Step 3. Perform an outward message passing on C_r in local JT T_i, such that C_r sends outward messages to all linkage hosts recursively.

An example of the partial belief update is illustrated in Figure 3. Suppose the last incoming message arrives over linkage tree L_0. The linkage hosts C_r, C_1 and C_3 have absorbed the message and the updated potentials of these clusters need to be propagated in the local JT T_0. As the modified potentials are only for clusters in the extended linkage host tree T_e, we first issue a partial inward pass in T_e for C_r to collect all inward messages. Next, outward messages are propagated from C_r to all the cluster in T_0, which brings all clusters to locally consistency with regard to the received incoming message.

By adopting the new message passing architecture and defining the concept of extended linkage host tree, we are able to restrict the propagation to a usually much smaller context in the local JT, and completely avoid full rounds of message passing during the LJF global propagation process.

4 Soundness

We first show that an inter-agent message calculated with Algorithm Cal_Single_MSG is consistent to the JT's local belief.

Proposition 1: Let T over the set of local variables N be a local JT of an agent A. Let L be T's linkage tree connecting to an adjacent agent A' over their d-sepset I. The extended linkage host tree is H, with C_r being the root cluster. For each linkage $Q \in L$, let $\Phi^*(Q)$ be the extended linkage potentials. After a call of Cal_Single_MSG to calculate an inter-agent message to A', we have

$$\prod_{Q \in L} \Phi^*(Q) = const \sum_{N \setminus I} \Phi(N)$$

where $\Phi(N)$ is the local belief calculated by multiplying N's initial belief and its all incoming messages except the one from A'.

Proof: First, consider the root cluster C_r in linkage host tree H as well as local JT T. After step 1 of Cal_Single_MSG, the messages except from A' are absorbed. Next, an inward propagation is performed in the local JT with regard to C_r as the root. Therefore, the potential associated with root cluster C_r defines the marginal of $\Phi(N)$ onto C_r. For C_r's corresponding linkage Q_{C_r} in L, $\Phi(Q_{C_r}) = const \sum_{N \setminus Q_{C_r}} \Phi(N)$. Next, consider all other linkage host clusters in H. Step 4 performs a partial outward JT message passing within H. After each $C_i \in H$ has received an outward message, the potential associated with C_i defines the marginal of $\Phi(N)$ onto C_i, and we have $\Phi(Q_{C_i}) = const \sum_{N \setminus Q_{C_i}} \Phi(N)$.

Since all clusters in H are consistent with the local belief and incoming messages, the correct linkage potentials can be obtained. Therefore, based on the definition of extended linkage potential, we have $\prod_{Q \in L} \alpha(Q) = const \Phi_L(I) = const \sum_{N \setminus I} \Phi(N)$.

□

Next, we show that after a call to Algorithm $Update\_Belief$, the local belief of an LJF local JF defines the marginal of joint system potential.

Proposition 2: Let F be the LJF of an MSBN over domain $\mathcal{N}$ and $Communicate\_Belief$ is performed in F. Let A be the agent of a local JT T over the local variables N, and let T_c be

the clusters of T. A has n neighboring agents $A_1, A_2, ..., A_n$. Let

$$\Phi(N) = \prod_{C \in T_c} \Phi(C) \prod_{i=1}^{n} \Phi(M_{A_i \to T}),$$

where $\Phi(C)$ is the potential initially assigned to a cluster C, and an incoming message received from T's one neighboring agent is denoted by $\Phi(M_{A_i \to T})$. Then,

$$\Phi(N) = const \sum_{\mathcal{N} \setminus N} \Phi_F(\mathcal{N}),$$

where $\Phi_F(\mathcal{N})$ represents the global belief of F over $\mathcal{N}$.

The most intuitive way to verify its validity is to view our global message passing in LJF as the Shenoy-Shafer message passings integrated with partial propagations for local updates. Proposition 1 shows that the message computed with Algorithm $Cal\ Single\ MSG$ is consistent with local belief. Similarly, Algorithm $Update\ Belief$ updates the local belief coherently given the last incoming message with a partial propagation. A proof to *Proposition 2* can be simply lifted from the correctness of Shenoy-Shafer JT global propagation to our LJF message passing architecture.

Based on the above two propositions, after a call to $Communicate\ Belief$ in an LJF is finished with all inter-agent messages delivered, each agent's local belief is locally as well as globally consistent w.r.t system belief.

5 Comparison

In an analysis with a comparison to other LJF inference architectures, we use the follow parameters:

- c: the maximum number of clusters in a local JT.
- d: the maximum number of clusters in an extended linkage host tree.
- s: the maximum number of adjacent agents.
- q: the cardinality of the largest cluster.

With Hugin-based LJF inference, each local JT cluster sends a message to, and receives a message from, each of its adjacent clusters during the local update. The time complexity of a single local update is thus $O(2c2^q)$ (Xiang, Jensen, and Chen 2006). Due to the fact that an agent must update local belief whenever a message is delivered from its adjacent agent, the local time complexity is $O(4cs2^q)$. We now consider the complexity of the local calculation with our architecture. During a call to calculate the first outgoing message, as only partial propagation is performed, the time complexity for message passing among local clusters is between $O(c + d)$ and $O((c + d)2^q)$, which is also the cost during the partial update on the last incoming message. The total cost for local calculation during a round of global belief update is between $O(2(c+d))$ and $O(2(c+d)2^q)$. As shown in Table 1, the local computation in our new architecture is clearly more efficient than the other methods, and particularly it does not depend on the topology of the network, e.g. the number of neighboring nodes.

	Local Cost
Hugin-based	$O(4cs2^q)$
Lazy-based	$O(4cs)$ - $O(4cs2^q)$
Our architecture	$O(2(c+d))$ - $O(2(c+d)2^q)$

Table 1: Comparison of time complexity for local computational cost.

6 Conclusion

Exact posterior calculation is one of the most important tasks of multi-agent probabilistic inference. The existing LJF-based algorithms typically extend the Hugin message passing and require an extensive amount of computation in LJF local JTs. In this paper, we presented a new global inference architecture based on the Shenoy-Shafer message passing scheme. We introduced new semantics for the LJF linkage trees, so that even the inter-agent messages of the same amount and content are passed, the local computation required in each local JT is optimized.

The improvement is realized first through the elimination of repeated local updates. Contrary to the existing methods that constantly maintain a transitional local consistency, our approach aims at one final consistent state at each local JT during the global propagation. Secondly, we conduct only partial propagation to handle all inter-agent messages. That is, no full-scale local message passing is ever invoked during the global propagation process.

The correctness of LJF global propagation is guaranteed in our new architecture. All local JTs are locally as well as globally consistent with the LJF joint system belief when the global propagation terminates. The complexity analysis has shown that the cost for message passing in an LJF local JT with our new architecture is much lower than the existing methods.

References

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

Jin, K. H., and Wu, D. 2008. Marginal calibration in multi-agent probabilistic systems. In *Proceedings of the 20th IEEE Int'l Conference on Tools with Artificial Intelligence*, 171–178.

Shenoy, P., and Shafer, G. 1986. Propagating belief functions using local computations. *IEEE Expert* 1(3):43–52.

Xiang, Y.; Jensen, F. V.; and Chen, X. 2006. Inference in multiply sectioned bayesian networks: Methods and performance comparison. In *IEEE Transaction on Systems, Man, and Cybernetics*, volume 36, 546–558.

Xiang, Y. 1995. Optimization of inter-subnet belief updating in multiply sectioned bayesian networks. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, 565–573. Morgan Kaufmann Publishers.

Xiang, Y. 2002. *Probabilistic Reasoning in Multuagent Systems: A Graphical Models Approach*. Cambridge.