Distributed Equipment Monitoring and Diagnosis with Multiply Sectioned Bayesian Networks

Y. Xiang and H. Geng
Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada S4S 0A2

Abstract

Multiply sectioned Bayesian networks (MSBNs) are an extension of Bayesian networks for flexible modeling and cooperative multiagent probabilistic inference. A large and complex equipment is modeled by a set of Bayesian subnets in a MSBN each of which corresponds to a natural component of the equipment. Each subnet forms the core knowledge of an autonomous M&D agent. Inference is performed in a distributed fashion while answers to queries are coherent with respect to probability theory. Recent advance in the MSBN theory shows that the coherence is not compromised even when internal knowledge of each agent is kept private from on another. Hence M&D systems can be integrated for very large equipments from agents built by different vendors.

We overview the MSBN framework and its features relevant to M&D tasks. We illustrate applications of MSBNs to M&D using constructed examples. We discuss several ongoing research issues regarding internet based MSBNs for M&D, agent upgrading through learning and handling of dynamic systems.

Introduction

Multiply sectioned Bayesian networks (MSBNs) are an extension of Bayesian networks for flexible modeling and cooperative multiagent probabilistic inference. A large and complex equipment is modeled by a set of Bayesian subnets in a MSBN each of which corresponds to a natural component of the equipment. Each subnet forms the core knowledge of an autonomous agent in monitoring and diagnosis (M&D) system. Inference is performed in a distributed fashion while answers to queries are coherent with respect to probability theory. Recent advance in the MSBN theory shows that the coherence is not compromised even when internal knowledge of each agent is kept private from others. Hence M&D systems can be integrated for very large equipments from agents built by different vendors with vendors' know-how protected. We have implemented most features of the framework in a JAVA based research toolkit, WEBWEAVR-III.

The focus of this paper is to overview the MSBN framework and present its features relevant to M&D tasks. We illustrate applications of MSBNs to M&D using constructed examples executed in WEBWEAVR-III. The paper is organized as follows: In the next section, we demonstrate how to model an equipment as a MSBN based multiagent system. We then present the issues of verification of the modeling and protection of privacy. The compilation of the model into its runtime representation is presented afterwards. The M&D capability of the multiagent system is then demonstrated with a case of trouble-shooting multiple faults. We discuss at the end several ongoing research issues regarding internet based MSBNs for M&D, agent upgrading through learning and handling of dynamic systems.

Modeling Equipment as MSBN

Bayesian networks (BNs) (Pea88; Jen96) provides a coherent and effective formalism for representation of uncertain knowledge for equipment monitoring and diagnosis (HBR95). The knowledge encoded in a BN includes the causal dependency of components (in terms of variables corresponding to devices and their input/output organized as a DAG structure), and the normal and faulty behavior of each device (in terms of the possible normal/faulty values of the corresponding variables and probability tables defined over these variables).

Such knowledge allows equipment monitoring to be performed since when an equipment functions normally, the observations will confirm the normal behavior, and the posterior probability (computed from the BN) of each device being normal will be high. Such knowledge allows diagnosis to be conducted as when observations differ from the normal behavior, those devices whose faulty behavior can best explain the observations will have high posterior probabilities of being faulty.

BNs, however, are limited by its centralized representation and inference. As the number of device/components of an equipment becomes larger, the components becomes more physically distributed, the vendors of components becomes more diverse, it becomes more difficult and inefficient to use centralized BNs for M&D.

The MSBN framework (XPB93; Xia96) provides an alternative to meet such needs. The key features of the framework are distribution of knowledge, distribution of inference, protection of knowledge ownership, and coherence of inference.

We shall use the digital circuit in Figure 1 to illustrate the application of MSBNs to M&D tasks. The principles are applicable to other types of equipments.

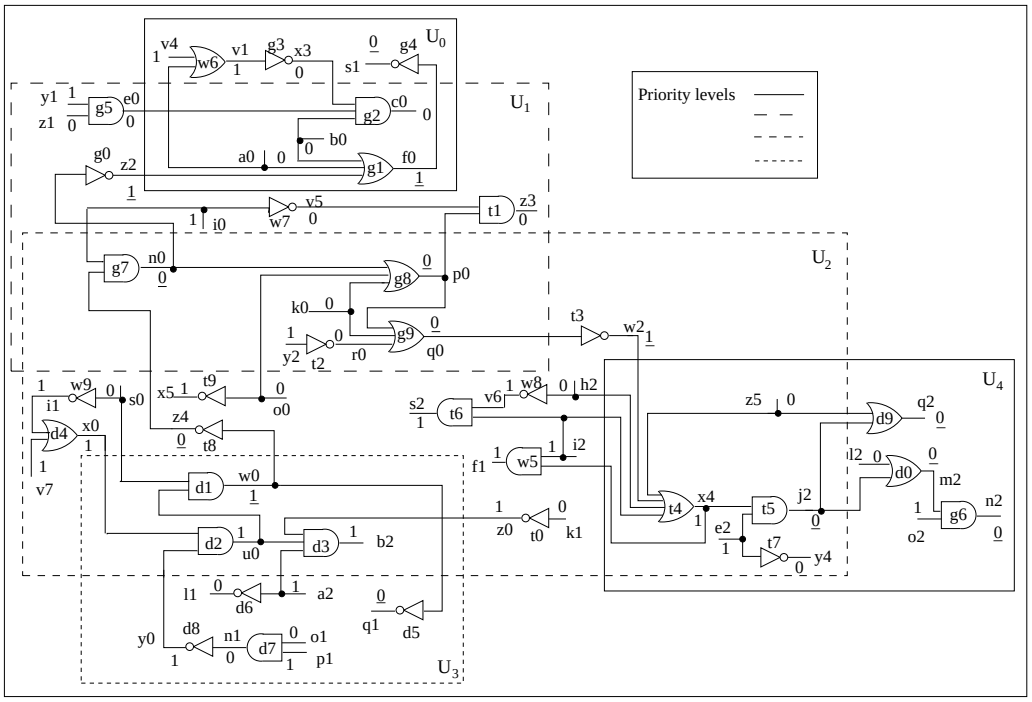

Figure 1: A digital circuit.

We assume that each equipment has a set of *external inputs* and a set of *external outputs*. In Figure 1, each input to a gate that is not the output of some gate is considered an external input, e.g., $y1$. Similarly, each output of a gate that is not the input of some gate is considered an external output, e.g., $c0$. If a signal is neither an external input nor an external output, then it is an *internal* signal, e.g., $x3$.

We assume that a complex equipment is structured into multiple components and each component consists of multiple devices. Each device is *physically* contained in a unique component.

The components of an equipment may be supplied by multiple vendors. A component vendor may not be willing to disclose the internal structure of the component. However, to allow the component to interface with other components, the vendor must disclose the information about the interface. Given the basic function of each component and its interface with others, an equipment vendor will be able to integrate components.

To monitor and diagnose an equipment, the equipment vender can use the framework of MSBNs to built a multiagent M&D system. Each agent is responsible for one component. It is supplied by the component vendor who encodes the internal of the component into the agent. An agent must also contain the information about the interface of the component with others. This information includes the input/output between the component and others. It may in general include the structure of some interfacing devices that do *not* physically belong to the component for which the agent is responsible. Hence a device may be *represented* in more than one agent.

We assume that the circuit is composed of five components $U_0, ..., U_4$ each of which is monitored by an agent A_i $(i = 0, ..., 4)$. In Figure 1, all devices enclosed in the box labeled U_0 are physically contained in the component. This means that the devices contained in both box U_0 and box U_1 are *not* physically contained in the component U_1, e.g., the gate $g1$. However, we assume that these devices are represented in both A_0 and A_1. In general, a device enclosed in two boxs is physically contained in the component signified by the box of the higher priority (see the priority levels box in the figure).

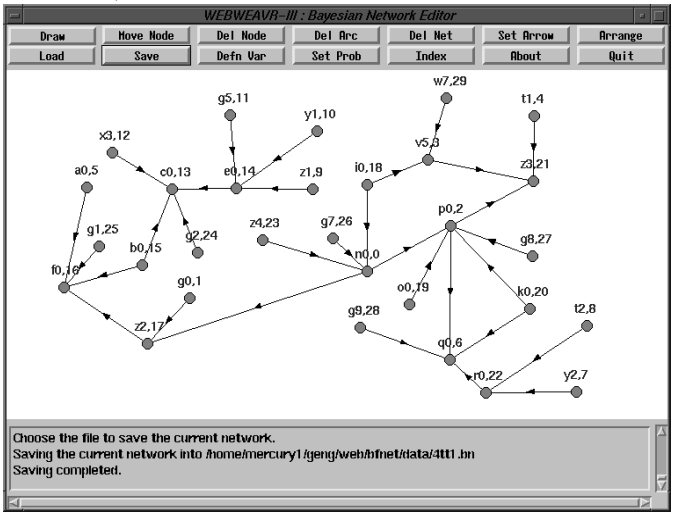

Figure 2: The subnet of agent A_1

The core knowledge representation in each agent A_i is a Bayesian subnet. The subnet of A_1 is shown in Figure 2 and that of A_2 is shown in Figure 3.

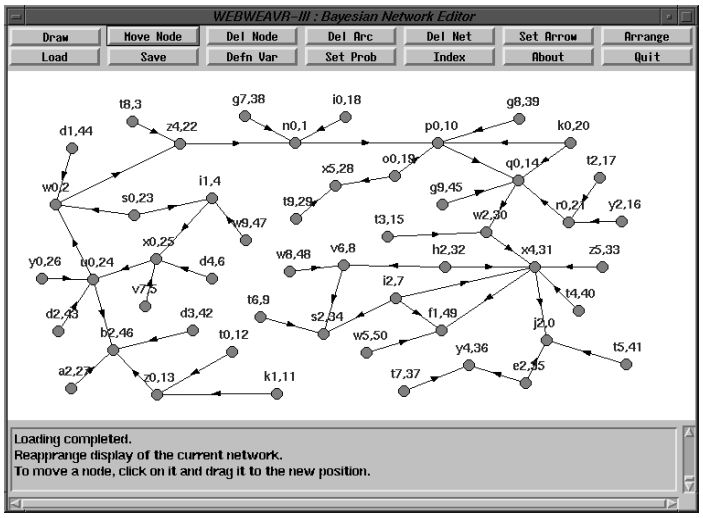

Figure 3: The subnet of agent A_2

Note that gates $g7$, $g8$, $g9$ and $t2$ are represented in both subnets.

We assume that each gate has a 0.01 probability of being faulty. A faulty AND gate can output correctly 20% of the time. A faulty OR gate, on the other hand, outputs correctly 70% of the time. A faulty NOT gate outputs correctly 50% of the time. These information are coded into the subnets.

Verification of Equipment Model

To ensure coherent distributed inference, the subnets that form a MSBN must satisfy a number of technical constraints. The formal presentation of these constraints can be found in (Xia96). Our discussion here will focus on their verification.

First, the subnets/agents must be organized into a (hyper)tree defined as follows:

Definition 1 *Let $G_i = (N_i, E_i)$ $(i = 0, ..., n-1)$ be n graphs. The graph $G = (\cup_i N_i, \cup_i E_i)$ is the* `union` *of G_is, denoted by $G = \sqcup_i G_i$.*

If for each i and j, $I_{ij} = N_i \cap N_j$ spans identical subgraphs in G_i and G_j, then G is `sectioned` *into G_is. I_{ij} is the* `separator` *between G_i and G_j.*

Definition 2 *(XJ99) Let $G = (N, E)$ be a connected graph sectioned into $\{G_i = (N_i, E_i)\}$. Let the G_is be organized as a connected tree H where each node is labeled by a G_i and each link is labeled by a separator such that for each i and j, $N_i \cap N_j$ is contained in each subgraph on the path between G_i and G_j in H.*

Then H is a `hypertree` *over G. Each G_i is a* `hypernode` *and each separator is a* `hyperlink`.

Each G_i corresponds to the structure of a subnet. The constraint ensures that no *circular* information passing is possible during multiagent inference. It can be verified using the *local covering* condition:

Definition 3 *Start with an empty graph (no node). Recursively add a DAG G_k to the existing union $\bigsqcup_{i=0}^{k-1} G_i$ such that the following holds:*

There exists G_i $(i < k)$ such that, for each G_j $(j < k; j \neq i)$, we have $I_{jk} \subseteq N_i$.

It appears that the verification of this condition requires the knowledge of the internal structure of each subnet. Research has shown that this is not true: The hypertree is assembled by the equipment vendor who has the knowledge of the interface of each component but not its internal structure. We shall call variables representing the interface of a component *public*, and call all other variables *private*. Since the private variables of each component are unique to the component, it is sufficient to use only the public variables in the verification of local covering. That is, the N_i in the above definition can be replaced by N_i' which stands for the union of all public variables/nodes in G_i. The verification can then be performed by cooperation among agents who propagate in the hypertree the relevant information on public nodes only according to Def.3.

Figure 4 shows a hypertree of the example system. Each hyperlink is labeled by the d-sepset. Only the public variables are shown in each oval representing an agent/subnet. Local covering is satisfied.

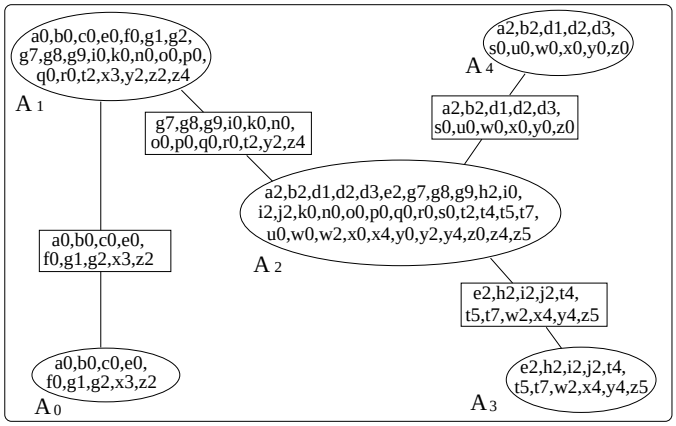

Figure 4: The hypertree of the circuit MSBN

Secondly, the structure of each subnet (called a *sub-DAG*) must be acyclic. This is the same structural constraint for conventional Bayesian networks. It is a local test and can be performed by individual agents. The subnet in each of Figures 2 and 3 is acyclic.

Third, the set of variables shared by each pair of subnets must form a d-sepset defined as follows.

Definition 4 *(XPB93) Let $G_i = (N_i, E_i)$ $(i = 0, 1)$ be two DAGs such that $G = G_0 \sqcup G_1$ is a DAG. The intersection $I = N_0 \cap N_1$ is a* `d-sepset` *for G_0 and G_1 if for every $x \in I$ with its parents $\pi(x)$ in G, either $\pi(x) \subseteq N_0$ or $\pi(x) \subseteq N_1$. Each $x \in I$ is called a* `d-sepnode`.

This constraint ensures that the interface of neighboring subnets allows sufficient information to be passed

between them during inference. It must be verified through cooperation of neighboring agents. The two agents must ensure that for each shared node x, it is not the case that each subnet has a parent of x not shared by the other subnet. Hence, the verification requires each agent reveals the parent set of each shared node, but no more information regarding the internal structure.

For example, the set of shared nodes between agents A_1 and A_2 is $\{g7, g8, g9, i0, k0, n0, o0, p0, q0, r0, t2, y2, z4\}$. Through cooperation of A_1 and A_2, it can be verified to be a d-sepset.

Finally, when all subnets are unioned (counting the shared nodes and links only once), the union graph must be acyclic. This is a direct extension of the structural constraint for conventional Bayesian networks. It can neither be verified locally by individual agents nor by pairwise testing between neighboring agents. An efficient algorithm has been developed (Xia98b), which requires cooperation of all agents through propagation of messages along the hypertree. The message from each agent must reveal whether a particular d-sepnode has any parent or child in the subnet. No more information about the internal structure of a subnet is needed.

Model Compilation

Once a MSBN based M&D system is verified, it will be compiled into a runtime representation where M&D inference can be performed more effectively. Each subnet is first *moralized* by connecting each pair of parents of each node and dropping directions. Messages on the *moral* links between d-sepnodes are propagated to neighbors to ensure consistent connections. No more information on the internal structure of each subnet needs to be revealed. Figure 5 shows the moral graph at agent A_1. The moral links are shows as grey and other links are black.

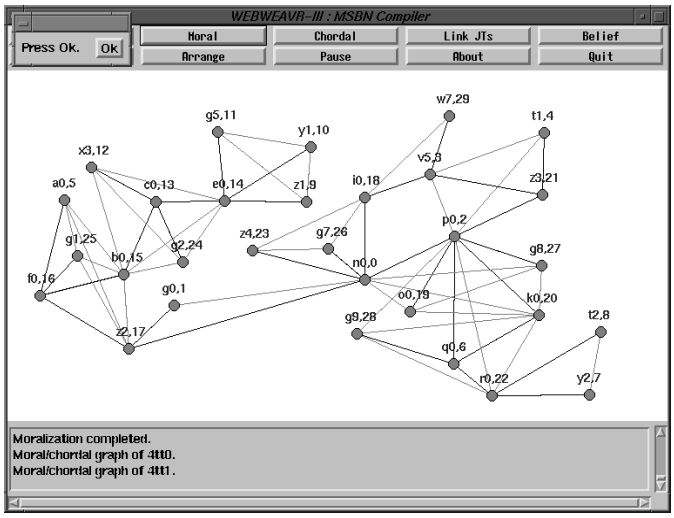

Figure 5: The moral graph at agent A_1

Then the graph union of the system is *triangulated* by local triangulation at each agent and propagation of added links (called *fill-ins* between d-sepnodes (Xia99)). No more information on the internal structure of each subnet needs to be revealed. Figure 6 shows the triangulated graph at agent A_1. Compare with Figure 5 to see added fill-ins.

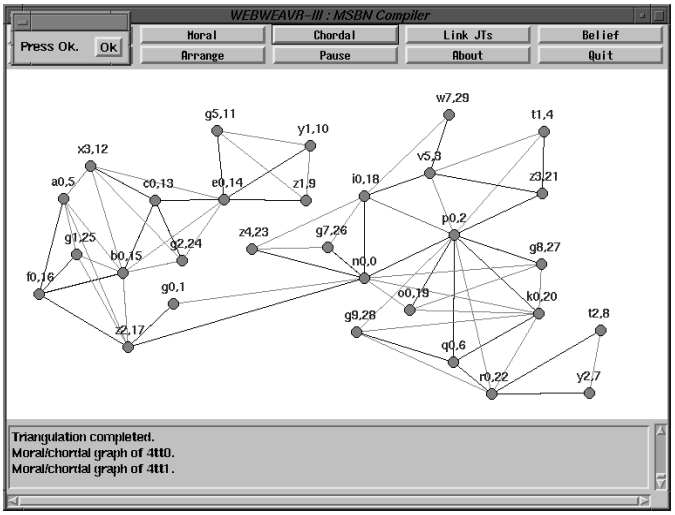

Figure 6: The triangulated graph at agent A_1

Each resultant local structure G_i is then converted to a *junction tree* T_i. Each node in T_i is labeled by a subset of nodes in G_i and is called a *cluster*. The clusters are so connected that the intersection of any two clusters are contained in each cluster on the unique path between them. The junction tree provides a tree structure that allows inference to be performed effectively using message passing among clusters (Jen96). Figure 7 shows the junction tree at agent A_1, where each cluster is labeled using the indexes of variables contained.

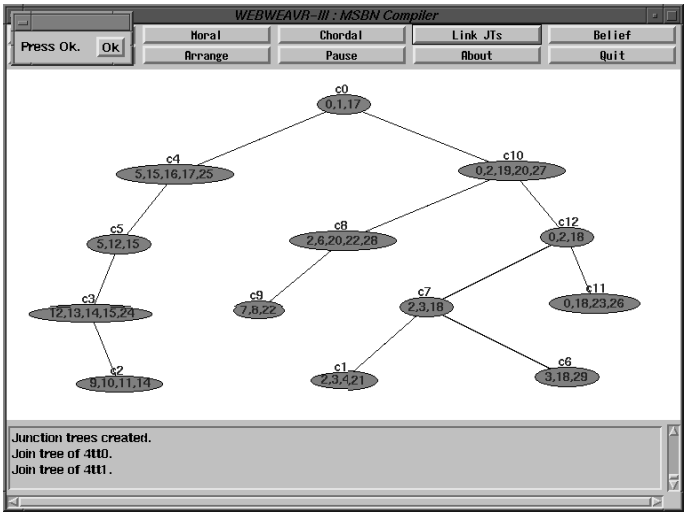

Figure 7: The junction tree at agent A_1

Finally, the d-sepset between each pair of agents is converted to a junction tree called a *linkage tree*. The linkage tree allows the belief on d-sepset to be decom-

posed into more compact representations and hence more efficient communication among agents. Figure 8 shows the linkage tree between agent A_1 and A_2.

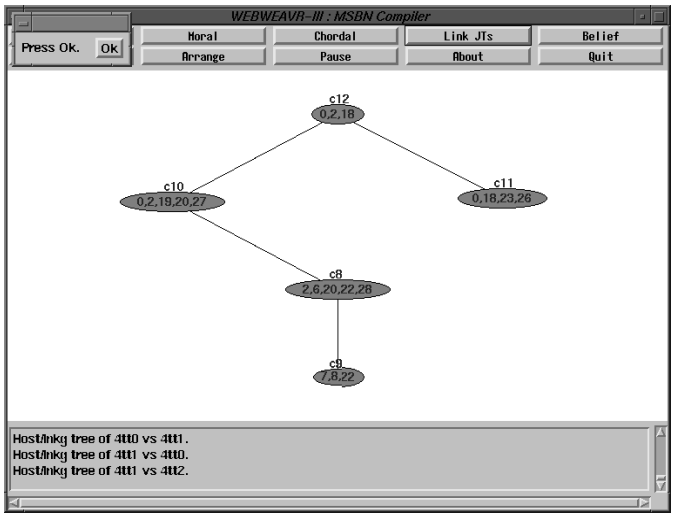

Figure 8: The linkage tree between agents A_1 and A_2.

The resultant representation of the MSBN is called a *linked junction forest*.

Monitoring and Diagnosis

M&D computation takes place in the linked junction forest, where each agent uses a junction tree as its internal representation and uses a linkage tree as the communication channel with a corresponding neighbor agent in the hypertree.

We assume the circuit in Figure 1 has two faulty gates $d1$ and $t5$ that produce incorrect outputs. The external inputs of the circuit are shown in the figure, and so are the outputs of all gates. Due to the faulty outputs of the two gates, outputs of some other gates are also affected. We have underlined each output that differs from the expected output value. Hence, Figure 1 defines a complete state of the circuit.

However, the state of the circuit is not entirely observable to the agents. We assume that the state of each gate is *not* observable. The external inputs and outputs of the circuit can be observed with low cost. Most internal outputs can be observed with high cost but some of them are also unobservable.

Autonomous inference without cooperation

First, we demonstrate the limitation of autonomous inference by individual agents without cooperation.

Consider agent A_4. Suppose that the output $x4$ of gate $t4$ is not observable. The belief of A_4 after all variables have been observed, except $x4$ and states of gates, is shown in Figure 9. The height of each histogram ranges from 0 to 1. The labels 0 and 1 represent logical values of inputs/outputs of gates. The labels b and g stand for states "bad" and "good" of gates.

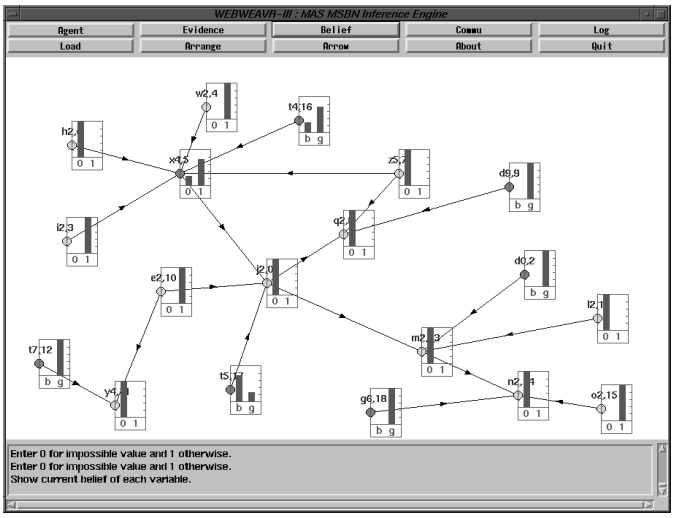

Figure 9: A_4 cannot decide if $t4$ or $t5$ is abnormal.

Since $x4$ is unobservable, although agent A_4 suspects that $t4$ or $t5$ might be faulty, it is quite uncertain (with belief 0.28 and 0.73 for each). Hence, A_4 is unable to decide replacement action with high certainty. We shall see in the following that by cooperating with other agents, A_4 can do much better.

Monitor and diagnose through cooperation

We assume that $x4$ and $w0$ are unobservable. Initially, each agent observes some local external inputs/outputs. A_0 observes $a0$, $b0$ and $s1$. A_1 observes $i0$, $k0$ and $z3$. A_2 observes $i2$, $s0$ and $b2$. A_3 observes $o1$ and $l1$. A_4 observes $i2$, $e2$, $o2$ and $n2$. Due to limited observation, most agents do not detect any abnormality yet. Figure 10 shows the belief of A_2.

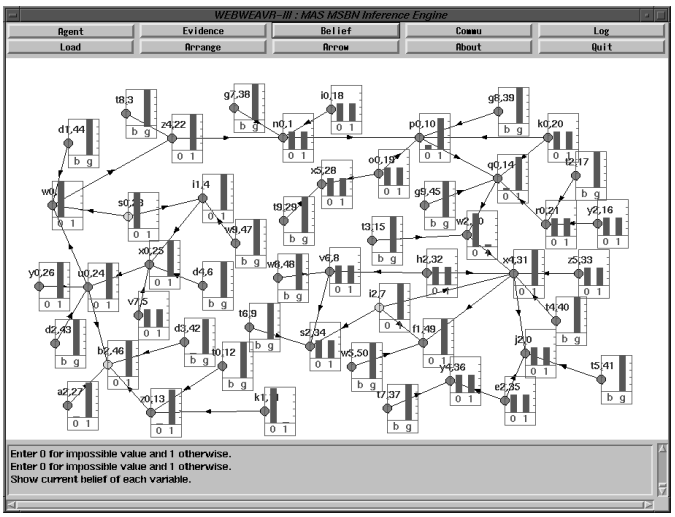

Figure 10: A_2 sees nothing wrong before communication.

Only A_4 detects that something is wrong (the probabilities of gates $t4$, $t5$, $d0$ and $g6$ being faulty are 0.10,

0.25, 0.19 and 0.49, respectively). It then initiates a communication.

During a communication, agents propagates their belief on d-sepsets along the hypertree by responding to requests from neighbors. After a communication, agents' belief will be consistent with all evidence accumulated in the system. The formal details can be found in (Xia96).

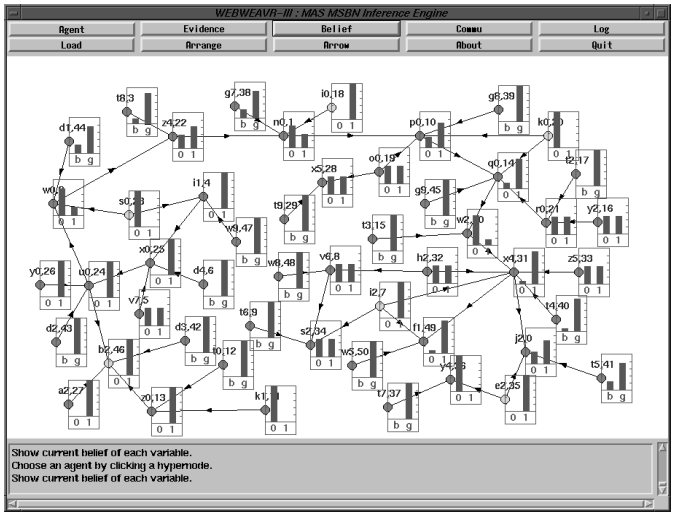

Figure 11: A_2 suspects gates $d1$, $t8$, $g7$, $t4$ and $t5$ after communication.

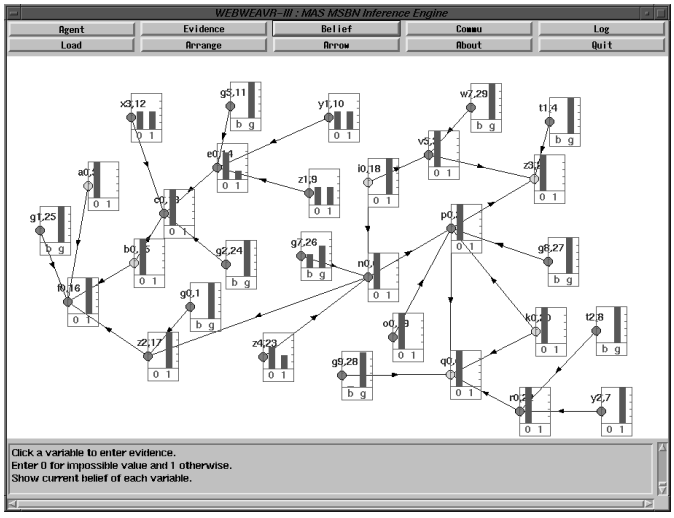

Figure 12: Agent A_1 makes an additional observation on $q0$.

After this communication, the deviation of circuit outputs from its expected values is detected by all agents. A_0's belief of gates $g1$ and $g4$ being faulty are 0.09 and 0.15. A_1's belief of gates $g0$, $g7$ and $g1$ being faulty are 0.15, 0.09 and 0.23. A_2 suspects abnormality in gates $d1$, $t8$, $g7$, and also gates $t4$ and $t5$ that physically belong to the subdomain of A_4 (Figure 11). A_3's belief of gate $d1$ being faulty is 0.23. A_4's belief of gates $t4$, $t5$, $g6$ and $d0$ being faulty are 0.10, 0.24, 0.49 and 0.19. It appears many gates are suspected but none is conclusive.

To further reduce the uncertainty, suppose that each agent makes one more observation. A_1 observes $q0$ and its belief is shown in Figure 12. It has now ruled out gates $g1$, $g0$ and $g9$, but still suspects $g7$.

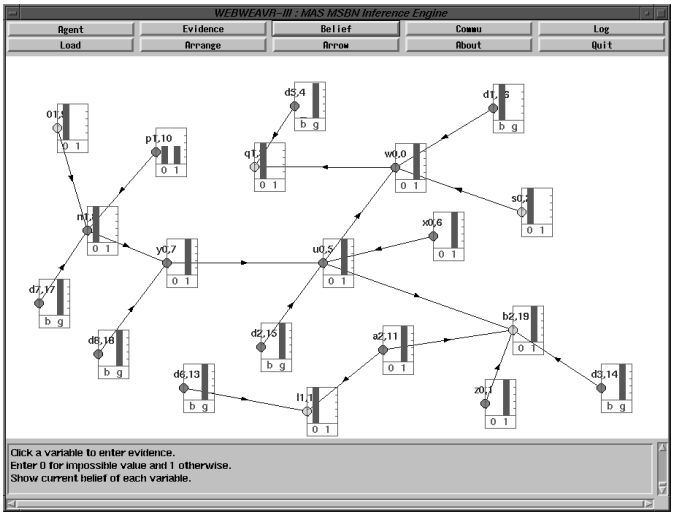

Figure 13: A_3 makes an additional observation on $q1$.

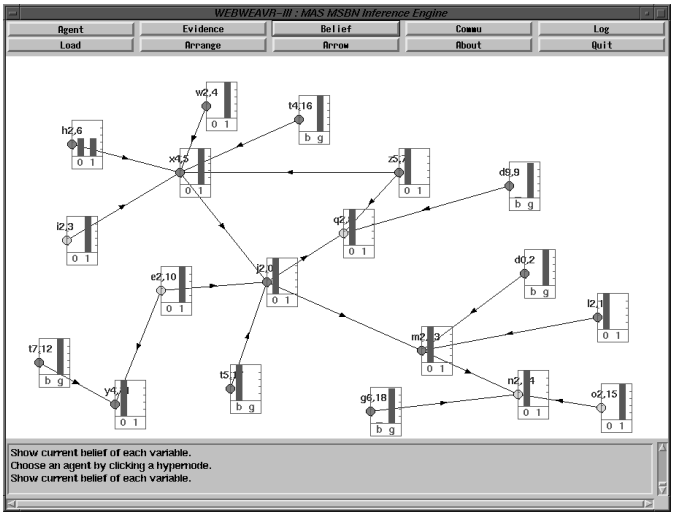

Figure 14: Agent A_4 knows that $t5$ is faulty after the second communication.

Figure 13 shows that A_3 is quite confident that $d1$ is faulty (belief 0.98).

After observing $c0$, A_0's belief on gates $g1$ and $g4$ does not change much. After observing $f1$, A_2 ruled out $t4$ with its belief on $t5$ being faulty increased to 0.27. Its uncertainty on gates $d1$, $t8$ and $g7$ are unchanged. A_4 observes $q2$ and rules out gates $d0$ and $g6$. It still suspect $t4$ (0.27) and $t5$ (0.72).

The unresolved uncertainty demands another communication. After that, agents A_0, A_1 and A_3 are cer-

tain that all gates in their physical domains are normal. A_2 is confident that only $d1$ is abnormal. A_4 is sure that gate $t5$ is faulty and everything else is normal (Figure 14).

The cooperation allows agents to converge their belief to high certainty even though some outputs of the gates are unobservable.

Research Issues

Internet based diagnosis

The agents in a M&D system do not have to run physically on site. That is, each M&D agent does not have to be located physically near the component that the agent is assigned to monitor. Instead, the agent may be running in a server at any convenient location distant from the component. Observations from the component can be transmitted to the agent through Internet or through a radio link if the component or the entire system is mobil.

The physical separation of a component and its monitoring agent allows the exploration of several possibilities. For example, the equipment owner does not have to own the M&D agent for each component. Instead, the component vendor owns the agent and uses it to provide the *on-line* M&D service.

A component vendor typically supplies for multiple equipments each owned by a difference owner, and hence will deploy one agent for each component object. All such agents, however, can be physically co-located in one or more servers at the location convenient to the component vendor. This allows the agents to be maintained and upgraded conveniently by the vendor. The cost of such a configuration is the transmission of observations from the components to the agent servers.

Utility and action

A common extension of BNs to include utility and actions is influence diagrams (Sha88). Influence diagrams assume predetermined possible sequences of observations/actions. Since agents are autonomous, it seems unreasonable to assume predetermined sequence of observations by multiple agents.

A straightforward extension of MSBNs to include utility and actions is the following: Each agent is given the utility of replacing each potentially faulty device (correctly or incorrectly). The action to take next is determined (myopically) by comparing the utility of replacing each device and following the principle of the maximum expected utility.

For information collecting actions (observations and communications), the utility may be determined through the cost and the value of information. An exact evaluation of the value of information for a particular action may be too expensive. An approximate evaluation can be performed relative to a single device currently suspected to be faulty by the local agent. The expected utility of the best action without the observation is computed, and so is the expected utility for actions with the observation taking into account all possible outcomes of the observation.

More accurate approximate evaluations may be performed relative to a group of local devices. It may also be desirable to take into account the values in other agents as some actions not valuable locally may be very valuable to other agents.

After a device believed to be faulty is replaced, all observations that depend on the state of the device are invalidated. The simplest way to reflect this is to let the agent request a rebooting of all agents. However, this may lose some observations on some agents which are not dependent on the newly replaced device. It would be desirable to develop less drastic updating that uses only local repairing instead of global rebooting.

Upgrading through learning

Effective M&D depends partially on the accuracy of numerical parameters embedded in each agent. Although a component's normal behavior should be well understood by the component vendor, its faulty behaviors and their relative frequencies may be unclear, especially when the component is new. The parameters embedded in the agent will likely be inaccurate.

As a component is deployed and monitored, observations on its behavior may be accumulated. These data may be used to improve the original parameters (LDLL90). Such data should be collected ideally from a large number of individuals of the device. This may be most conveniently performed if all the M&D agents for the device are controlled by the component vendor as we outlined above.

Extending to dynamic systems

Many equipments' states change over time. The current theory of MSBNs does not handle time explicitly. We consider the limit of the current theory in monitoring such dynamic systems.

Many faults of such systems can be identified without referring to its history. It is sufficient to identify these faults by using only a single snap shot of the system state at the right time. A MSBN based MAS can be used to diagnose such faults. Agents make observations and communications regarding only the system state at a particular instant. They repeatedly perform such activities for each instant *independently* of other instants.

Other faults can only be identified by looking into the history of the system states. Depending on the memory length k (how far into the past the current state depends on) of the system, we classify them into those of *short* memory length and those of *long* memory length.

When the equipment has a short memory length (k is small), it may still be feasible to model the equipment as a MSBN. For each dynamic variable, up to k instances can be created each representing the value of the variable at a particular instant. Agents reason treating k successive instants as a unit. The units may or may not overlap. In either case, agents repeatedly

perform their activities for each unit *independently* of other units.

As k increases, the representation and computation will become too complex to be feasible. Hence for equipments of long memory length, the current theory of MSBNs must be extended in order to be applicable. Under the single agent paradigm, the counterpart of a static BN is the *dynamic* Bayesian networks (DBNs). Although inference in MSBNs are *exact*, there are evidence (Xia98a) that approximation is unavoidable when extending MSBN in a similar fashion.

Conclusion

We presented the MSBN framework in the context of equipment monitoring and diagnosis. We demonstrated that modeling, verification, compilation and M&D inference can all be performed distributed in a multiagent setting. The belief of agents through such inference is coherent with respect to the probability theory. The disclosure of the internal structure of each agent's knowledge base is minimum. Hence, the framework provides a powerful formalism for integrating multiple agents from difference vendors into a coherent M&D system, while protecting the know-hows of component vendors.

We have implemented most of the features demonstrated in a JAVA based toolkit WEBWEAVR-III (freely available from the first author's homepage). Promising directions are currently being explored to extend the framework to include utility and actions, and to handle dynamic systems. We are also actively seeking industrial partners in applying the framework to equipment M&D.

Acknowledgements

This work is supported by the Research Grant OGP0155425 from the Natural Sciences and Engineering Research Council (NSERC), and a grant from the Institute for Robotics and Intelligent Systems in the Networks of Centres of Excellence Program of Canada.

References

D. Heckerman, J.S. Breese, and K. Rommelse. Decision-theoretic troubleshooting. *Communications of the ACM*, pages 49–57, 1995.

F.V. Jensen. *An introduction to Bayesian networks*. UCL Press, 1996.

S.L. Lauritzen, A.P. Dawid, B.N. Larsen, and H.G. Leimer. Independence properties of directed Markov fields. *Networks*, 20:491–505, 1990.

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

R.D. Shachter. Probabilistic inference and influence diagrams. *Operations Research*, 36(4):589–604, 1988.

Y. Xiang. A probabilistic framework for cooperative multi-agent distributed interpretation and optimization of communication. *Artificial Intelligence*, 87(1-2):295–342, 1996.

Y. Xiang. Temporally invariant junction tree for inference in dynamic Bayesian network. In R.E. Mercer and E. Neufeld, editors, *Advances in Artificial Intelligence*, pages 363–377. Springer, 1998.

Y. Xiang. Verification of dag structures in cooperative belief network based multi-agent systems. *Networks*, 31:183–191, 1998.

Y. Xiang. Cooperative triangulation in MSBNs without revealing subnet structures. *Networks*, 1999. under revision.

Y. Xiang and F.V. Jensen. Inference in multiply sectioned Bayesian networks with extended Shafer-Shenoy and lazy propagation. Submitted for publication, 1999.

Y. Xiang, D. Poole, and M. P. Beddoes. Multiply sectioned Bayesian networks and junction forests for large knowledge based systems. *Computational Intelligence*, 9(2):171–220, 1993.