Probabilistic Reasoning in Dynamic Multiagent Systems

Xiangdong An
School of Computer Science
University of Waterloo
Waterloo, ON N2L 3G1, Canada
Email: xan@cs.uwaterloo.ca

Yang Xiang
Dept. of Comp. and Info. Sci.
University of Guelph
Guelph, ON N1G 2W1, Canada
Email: yxiang@cis.uoguelph.ca

Nick Cercone
Faculty of Computer Science
Dalhousie University
Halifax, NS B3H 1W5, Canada
Email: nick@cs.dal.ca

Abstract

Probabilistic reasoning with multiply sectioned Bayesian networks (MSBNs) has been successfully applied in static domains under the cooperative multiagent paradigm. Probabilistic reasoning in dynamic domains under the same paradigm involves several issues. This paper proposes an approach to address these issues. Intuitively, observation on current state plays a more important role in the reasoning of the current state than remote historic information. Based on the intuition, we model the entire domain for a period of time into an MSBN and then reason about the state of the dynamic domain period by period exactly. In reasoning the state of a suspected entity, we compute and observe an observable Markov boundary of the entity. This makes observation more efficient and relevant. In MSBNs, an observable Markov boundary of a node may span across all Bayesian subnets. We present an algorithm for cooperative computation of an observable Markov boundary of a set of nodes in MSBNs without revealing subnet structures. Preliminary experiments show the approach works well on our simulated multiagent dynamic domains.

Introduction

Bayesian networks (BNs) (Pearl 1986; 1988; Castillo, Gutierrez, & Hadi 1997; D'Ambrosio 1999) and dynamic Bayesian networks (DBNs) (Dean & Kanazawa 1988; 1989; Kjaerulff 1992; Murphy 2002) have been widely accepted as the respective graphical models for uncertain reasoning in static domains and dynamic domains under the single agent paradigm. Multiply sectioned Bayesian networks (MSBNs) (Xiang, Poole, & Beddoes 1993; Xiang & Lesser 2000; Zhang, Tian, & Lu 2001; Xiang 2002) have been successfully applied in static domains under the cooperative multiagent paradigm. This paper introduces cooperative multiagent probabilistic reasoning into distributed dynamic domains. Probabilistic reasoning in dynamic multiagent environments can be applied in many areas such as distributed monitoring and troubleshooting, workflow and business process management (e.g. logistics, manufacturing), information retrieval and management (e.g. cooperating internet agents), electronic commerce, and virtual environments (e.g. multi-robot environments).

It has been shown that the decomposed representation of joint probability distributions (JPDs) may not be well supported in dynamic domains (Boyen & Koller 1998; Xiang 1999; Bilmes & Bartels 2003). It is also demonstrated that the spatial distribution of the multiagent systems conflicts with temporal message passing for dynamic multiagent probabilistic reasoning (Xiang 2002; An 2003). These issues do not allow each agent to obtain historic information separately and benefit from each other's knowledge up to the relevant history. Intuitively, observation on current state plays a more important role in the reasoning of the current state than remote historic information. Based on the intuition, we model the entire domain for a period of time into an MSBN and then reason the state of the dynamic domain using the MSBN period by period. By this way, the state of the domain in each period can be reasoned about exactly. Regarding how long the period should be, there must be a trade-off between taking advantage of historic information and reducing computational complexity. In reasoning the state of an entity, we compute and observe an observable Markov boundary of the entity. This makes observation more efficient and relevant. Since an observable Markov boundary of a node may span across all Bayesian subnets of an MSBN, we provide an algorithm to cooperatively compute an observable Markov boundary without revealing private subnet structures. The approach is tested on our simulated dynamic multiagent domains.

The rest of the paper is organized as follows. Section 2 overviews MSBNs. Section 3 proposes to extend MSBNs to dynamic domains for cooperative multiagent probabilistic reasoning. Issues related with the uncertain reasoning using the extended MSBNs are simply reviewed. Section 4 proposes an approach to deal with these issues. Section 5 examines Markov boundaries in MSBNs and provides an algorithm to compute an observable Markov boundary of a set of nodes in an MSBN. Section 6 demonstrates the approach over a simulated dynamic domain. Conclusion is presented in Section 7.

Multiply Sectioned Bayesian Networks

A multiply sectioned Bayesian network (MSBN) is a knowledge representation formalism for cooperative multiagent uncertain reasoning (Xiang 1996). The framework allows a large domain to be modeled modularly and the effective inference to be performed distributedly while maintaining the coherence (Koller & Pfeffer 1997).

An MSBN consists of a collection of Bayesian subnets

each of which encodes an agent's knowledge on a subdomain. All agents are organized into a junction tree (JT) (Lauritzen & Spiegelhalter 1988; Castillo, Gutierrez, & Hadi 1997) structured communication graph called *hypertree* where each *hypernode* represents an agent and each *hyperlink*, also called an *interface*, represents a pathway for direct inter-agent communication. An interface is a set of shared nodes between any two adjacent agents, which is analogous to a separator in the single agent JT. The union of all agents is a larger Bayesian network (BN).

Definition 1 *Let $G = (V, E)$ be a connected graph sectioned into subgraphs $\{G_i = (V_i, E_i)\}$. Let these subgraphs be organized into a tree Ψ where each node, called a* hypernode*, is labeled by G_i and each link between G_i and G_j, called a* hyperlink*, is labeled by the interface $V_i \cap V_j$ such that for each pair of nodes G_l and G_m, $V_l \cap V_m$ is contained in each subgraph on the path between G_l and G_m. The tree Ψ is called a* hypertree *over G.*

Therefore, if each hypernode G_i is considered a cluster over V_i, the hypertree Ψ is a junction tree which satisfies the *running intersection property* (Castillo, Gutierrez, & Hadi 1997).

Definition 2 *Let G be a directed graph such that a hypertree over G exists. A node x contained in more than one subgraph with its parents $\pi(x)$ in G is a* d-sepnode *if there exits a subgraph that contains $\pi(x)$. An interface I is a* d-sepset *if every $x \in I$ is a d-sepnode.*

On d-sepset, we have the following conclusion (Xiang 2002).

Theorem 3 *Let Ψ be a hypertree over a directed graph $G = (V, E)$. For each hyperlink I which splits Ψ into two subtrees over $U \subset V$ and $W \subset V$ respectively, $U \setminus I$ and $W \setminus I$ are independent given I if and only if the hyperlink I is a d-sepset.*

Definition 4 *A hypertree MSDAG $G = \bigcup_i G_i$, where each $G_i = (V_i, E_i)$ is a directed acyclic graph (DAG), is a connected DAG such that there exists a hypertree over G and each hyperlink is a d-sepset.*

An MSBN consists of its structure as defined above as well as its numerical probability distributions. The following definition of an MSBN specifies how the numerical distributions are associated with the structure.

Definition 5 *An* MSBN *M is a triplet $(V, G, \mathcal{P})$. $V = \bigcup_i V_i$ is the total universe where each V_i is a set of variables, called a* subdomain*. $G = \bigcup_i G_i$is a hypertree MSDAG where nodes of each subgraph G_i are labeled by elements of V_i. Let x be a variable and $\pi(x)$ be all parents of x in G. For each x, exactly one of its occurrences (a G_i containing $\{x\} \cup \pi(x)$) is assigned $P(x|\pi(x))$, and each occurrence in other subgraphs is assigned a unit constant potential. $\mathcal{P} = \prod_i P_i$ is the JPD where each P_i is the product of the potentials associated with nodes in G_i. Each triplet $S_i = (V_i, G_i, P_i)$ is called a* subnet *of M. Two subnets S_i and S_j are said to be* adjacent *if G_i and G_j are adjacent in the hypertree.*

Figure 1 illustrate an MSBN with a trivial MSDAG, where each box represents the DAG of one agent. The corresponding hypertree is depicted in Figure 2. The interfaces between G_0 and G_1, between G_1 and G_2, and between G_1 and G_3 are $\{x, z\}$, $\{x, y\}$ and $\{u, v\}$, respectively. All these interfaces are d-sepsets because $\pi(x) \subseteq V_1$, $\pi(z) \subseteq V_0$, $\pi(y) \subseteq V_2$, $\pi(u) \subseteq V_1$ and $\pi(v) \subseteq V_1$, where V_i denotes the set of nodes in G_i. If an interface is not a d-sepset, the interface won't render the subdomains in the two induced subtrees conditionally independent. Therefore, the hypertree used to organize all Bayesian subnets in an MSBN is actually a junction tree to facilitate message passing among agents over interfaces.

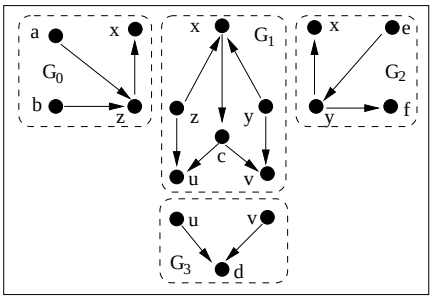

Figure 1: An MSBN with a trivial MSDAG.

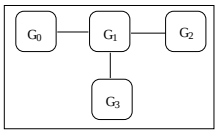

Figure 2: The hypertree for the MSDAG in Figure 1.

MSBNs provide a framework for probabilistic reasoning in cooperative multi-agent distributed interpretation systems. In an MSBN, each agent holds its partial perspective of a large problem domain, accesses a local evidence source (sensors), communicates with other agents infrequently, reasons with local evidence and limited global evidence, and answers queries or takes actions. Agents are cooperative in that the joint system belief is well defined which is identical to each agent's belief within its subdomain and supplemental to the agent's belief outside the subdomain (Xiang 1996). Even though multiple agents may acquire evidence asynchronously in parallel, the communication operations of MSBNs ensure that the answers to queries from each agent are consistent with all local evidence acquired so far and are consistent with all evidence gathered in the entire system up to the last communication.

Dynamic Multiagent Probabilistic Reasoning

Dynamic MSBNs

Multiply sectioned Bayesian networks have been successfully applied as a graphical model for probabilistic reasoning in static domains under the distributed and multiagent

paradigm. We propose to introduce probabilistic reasoning into distributed dynamic multiagent domains by extending MSBNs.

We propose at each time instant a dynamic multiagent domain is modeled with an MSBN. All MSBNs over all time instants are called a *dynamic MSBN*. The MSBN over one time instant is called a slice of the dynamic MSBN. In a dynamic MSBN, there exists a DBN corresponding to each subdomain where the uncertain knowledge at each time instant is represented by a BN. In each subdomain, the interface between two consecutive Bayesian subnets is called a *slice subinterface* and all slice subinterfaces between two consecutive slices form a *slice interface* of the dynamic MSBN.

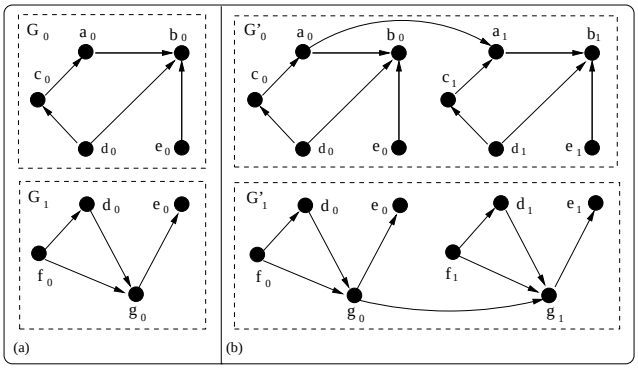

Figure 3: An MSBN extended over a dynamic domain: (a) The MSBN over two subdomains G_0 and G_1 before extension; (b) The extended MSBN over two time instants 0 and 1.

Figure 3 shows a two agent MSBN extended over a dynamic domain. The structure of one slice of the dynamic MSBN is shown in (a) and the first two consecutive slices of the extended MSBN are shown in (b). The temporal dependencies are signified by the arcs (a_0, a_1) and (g_0, g_1) respectively. A DBN is formed in each subdomain.

Now we have a model to represent the distributed knowledge in dynamic multiagent systems. However, agents have difficulty to obtain historic information separately and benefit from each other's knowledge up to the relevant history.

Issues

Decomposition Issue It has been shown that the decomposed representation of joint probability distributions (JPDs) may not be well supported in dynamic domains. Due to temporal dependencies, the variables in one slice may become fully correlated very early in time (Boyen & Koller 1998). In general, the state of a dynamic domain can be reasoned about using a JT template. The JT template should be obtained by triangulating the DBN in an temporal order (e.g. by always eliminating earlier nodes first) (Xiang 1999; Darwiche 2001; Bilmes & Bartels 2003). In a JT template such obtained, the slice interface members would be complete (Xiang 1999; Bilmes & Bartels 2003).

All these would be true when probabilistic reasoning are applied in dynamic multiagent domains. That is, all variables in the domain may become fully correlated very early in time and slice interface members may become complete if triangulation is done chronologically.

Distribution Issue We could not reason about a large dynamic domain over an unbounded time period directly by treating the domain as a static domain. Ideally, we would like each agent to maintain its belief on its subdomain for the current time instant and all agents to be able to benefit from each other's knowledge up to the relevant history. However, each agent cannot propagate its belief on its subdomain from one time instant to next time instant separately, even when Markovian property holds in the dynamic domain as a whole (An 2003). Actually, the message passing separately in each subdomain constructs the message passing in non-degenerate cycles (loopy belief propagation). It has been shown coherent belief updating cannot be achieved in a cluster graph with non-degenerate cycles, no matter how message is passed (Pearl 1988; Murphy, Weiss, & M.I.Jordan 1999; Xiang & Lesser 2000; Crick & Pfeffer 2003). Intuitively, message passing separately in each subdomain loses the dependency information among all messages passed over the slice subinterfaces. We have difficulty to pass exact probabilistic message distributedly. Further more, in each slice of a dynamic MSBN, the agent interfaces may not separate the two subdomains they connect (Xiang 2002).

This shows that exact multiagent probabilistic reasoning over unbounded time periods cannot be achieved by maintaining agent belief over a finite time. This further implies that the spatial distribution of the multiagent systems conflicts with the temporal dependency of the dynamic domains for the probabilistic reasoning.

Local Inference

In dynamic multiagent domains, agents have difficulty to obtain historic information separately and benefit from each other's knowledge up to the relevant history for probabilistic reasoning. Both history information and the observation of the current state are helpful in the reasoning of the current state of a dynamic domain, but most times observation of the current state may play a more important role. For example, historical health information is useful in current medical therapy, but observation on current health state would be more helpful. Given current observation, some history information may become independent of the current state. Especially, it is possible belief revision from old unreliable observation can be corrected by new observation (Friedman & Halpern 1996; Boutilier, Friedman, & Halpern 1998).

Hence, we propose to tackle all those issues by modeling each k-slices of a dynamic MSBN into an MSBN M. For each new k-instant period, the initial prior belief of the first k-instant period is taken. There is not any message passing between two consecutive periods. The state of each period can be reasoned about exactly by the MSBN model M. Then the state of the dynamic domain can be reasoned about period by period. Therefore, in each period, there is no need for separate forward message passing in each sub-

domain. Between any two adjacent subdomains, the agent interface in M would include all corresponding interfaces over the period. For example, Figure 3 (b) shows a dynamic MSBN over two time instants 0 and 1. We can model the two slices of the dynamic MSBN into an static MSBN whose two Bayesian subnets are G'_0 and G'_1. The interface between two subdomains would be $\{d_0, e_0, d_1, e_1\}$. The "separation" property of such agent interfaces could be affected by temporal dependencies between current period and previous periods. We ignore the message between two consecutive periods. Intuitively, the remote history information could be ignored. It has been shown that, in a static domain, the influence of evidence would become weakened over distance (de Campos & Fernandez-Luna 2002). Corresponding to dynamic domains, the longer the time, the farther the distance, the weaker the influence. The influence of temporal dependencies between two consecutive periods on "separation" property of the agent interfaces in one period MSBN could become weakened over time and be ignored.

By this approach, we inherit information from the most recent history and ignore effects of the remote history. We rely more on the observation than on the remote history.

Markov Boundary in MSBNs

Due to the limitation of bandwidth, we may not be able to observe all observable variables. We may have to observe most relevant variables. MSBNs are usually used to model large and complex domains, so the number of variables in an MSBN should be large and the topology of an MSBN would be complex. It should not be very easy to find variables most relevant to a suspected entity. For example, it is possible we may choose to observe a variable which appears to be relevant to the state of another variable but actually not (Xiang 2002).

In a BN, a Markov boundary $B(\alpha)$ of a variable $\alpha \in U$ is a minimal subset $S \subseteq U$ of variables that makes the state of α independent of any variables outside the boundary. That is, it makes $I(\alpha, S, U \setminus S \setminus \{\alpha\})$ hold. Hence, its Markov boundary members would be the most relevant and helpful variables to observe in reasoning the state of a variable. For a variable α in a BN, one of its Markov boundaries is the union of its parents, its children and the parents of its children.

Markov Boundary over a Node

For a node α in an MSBN, its Markov boundary members may exist in one Bayesian subnet, or in several Bayesian subnets.

If α is a private node which uniquely exists in only one Bayesian subnet N, its Markov boundary members could be private nodes in N or shared (d-sepset) nodes with any other than N subnets. For example, in Figure 3 (b), node g_1 in subnet G'_1 is a private node. Its Markov boundary contains nodes d_1, e_1, f_1 and g_0 where d_1 and e_1 are shared nodes. Nevertheless, they must all exist in the Bayesian subnet N where α exists. This is because both α's parents and children won't be private nodes in Bayesian subnets other than N (they are at most shared nodes with N's adjacent subnets), and its children won't have any parents in subnets other than N but not shared with N (Theorem 3). Hence, in this case, the agent over N can compute α's Markov boundary in N and then informs its adjacent agents of any shared nodes which are boundary members. For an entity, we call the subdomain where it physically exists its *host subdomain* and corresponding agent *its host agent*. If a node is a private node, the unique subdomain where it exists is its host subdomain and corresponding agent its *host agent*. If a node is a shared node, the subdomain where it physically exists is its host subdomain and corresponding agent its host agent. A node can only be observed by its host agent.

If α is a shared node, the nodes in its Markov boundary could be private nodes or shared nodes. They may exist in several of all the Bayesian subnets of an MSBN M. All its parents in M should exist in one subdomain. Its children and the parents of its children could exist across all subdomains of M. However, any of its Markov boundary members should appear in one subdomain containing α. If any β of its Markov boundary members does not appear in any subdomains that contain α, then there does not exist a subdomain that contains both α and β. This implies there exists at least one extra node on any path between α and β. It is not true if β is a parent or a child of α. If β is a parent of a child of α, then there should be a subdomain containing both α and β (d-sepset). For example, e_1 in Figure 3 is a shared node between two adjacent subnets. We start the computation of its Markov boundary from subnet G'_1 because G'_1 contains all parents of e_1. Hence, g_1 would be one of its Markov boundary members. The child b_1 of e_1 and the parents a_1 and d_1 of b_1 exist in subnet G'_0 where d_1 is a shared node. Therefore, each of the agents over subdomains containing α should list all parents and all children of α and all parents of α's children as the Markov boundary members of α. Then all shared Markov boundary members could be distributed to corresponding adjacent agents for observation.

Observable Markov Boundary over a Set of Nodes

Sometimes, all of a set of neighbouring nodes related with a suspected entity may not be observable. We may have to compute the Markov boundary over the set of nodes. For example, in Figure 3 (b), initially we want to compute the Markov boundary of node g_1. The Markov boundary would be $\{d_1, e_1, f_1, g_0\}$ if observabilities are not considered. However, if we assume g_0 is unobservable, we may have to further compute the Markov boundary of both g_0 and g_1. If all other nodes are assumed to be observable, then the Markov boundary of $\{g_0, g_1\}$ would be $\{d_0, e_0, f_0, d_1, e_1, f_1\}$. All members in the boundary are the most relevant variables we could observe to determine the state of g_1. Sometimes, Markov boundary may expand outside a subnet in an MSBN. For example, in Figure 3, we assume f_0 is unobservable and its Markov boundary over the MSBN would be $\{d_0, g_0\}$ if they are observable. However, if we assume d_0 is unobservable, the Markov boundary would expand outside G'_1 and becomes $\{g_0, c_0, b_0, a_0, e_0\}$. In particular, if we further assume a_0, a_1, c_1 and d_1 are unobservable, the Markov boundary would expand back to G'_1 to include f_1 and g_1. Therefore, a Markov boundary of a node could be expanded over and over again until it becomes

stable.

Computation of the Observable Markov Boundary in MSBNs

Algorithms 1, 2, 3, 4, 5 and 6 are a suite of algorithms to compute observable Markov boundary of a set of unobservable nodes H in an MSBN M of n subnets. In these algorithms, we say a node *observable* in whole domain if it can be observed by one agent, although an observable node can only be observed by its host agent. In particular, each agent should keep a list of Markov boundary members of H it can observe in its subdomain, called a *partial Markov boundary* of H in the subdomain, for observation in inference. Algorithm 3 computes Markov boundary of a set of nodes in a Bayesian subnet. Algorithm 2 expands the boundary across subnets until the boundary becomes stable. Algorithm 1 makes some initialization and calls Algorithm 2. Algorithm 4 calls Algorithms 5 and 6 to distribute all shared Markov boundary members recursively to corresponding agents. Algorithm 3 marks any node checked to ensure each node will be checked at most once. Hence, the computational complexity of the algorithm would be $O(mn)$, where m is the number of nodes in the largest subnet of M.

Algorithm 1 (ComputeMarkovBoundary) *Let H be a set of unobservable nodes in an MSBN M. Let S_1, S_2, ..., S_n be n subnets of M over subdomains $V_1, V_2, ..., V_n$, respectively. Let A_1, A_2, ..., A_n be the corresponding agents over the subnets. Let $B_1, B_2, ..., B_n$ be the respective partial Markov boundaries of α in subnet $S_1, S_2, ..., S_n$, respectively.*

First, the system coordinator does the following:

for each agent A_i ($i = 1, 2, ..., n$), do
  set $B_i = \emptyset$;
  create a set T_i;
  set $T_i = V_i \cap H$;

Then, the system coordinator calls ExpandMarkovBoundary;

Algorithm 2 (ExpandMarkovBoundary) *Denote all adjacent agents of an agent A_c by $A_{k1}, A_{k2}, ..., A_{kc}$. Denote the set of unobservable nodes reached by each agent in ComputePartialMarkovBoundary by $Z_1, Z_2, ..., Z_n$.*

Each A_c of all agents ($A_1, A_2, ..., A_n$) whose corresponding $T_c \neq \emptyset$ does the following:

run ComputePartialMarkovBoundary;
for each adjacent agent A_i ($i = k1, k2, ..., kc$) of A_c, do
  pass the shared nodes in $Z_c \cap V_i$ to T_i of A_i;
  call A_i to remove any marked nodes from T_i;

if all T_i's ($1 \leq i \leq n$) are $\emptyset$'s, do
  call UnifyMarkovBoundary;
  halt;
otherwise restart the algorithm;

Algorithm 3 (ComputePartialMarkovBoundary) *Let T_i be a set of unobservable nodes in subnet S_i, Agent A_i does the following to compute observable partial Markov boundary of T_i in S_i:*

mark all nodes in T_i and set $Z_i = T_i$;
while T_i is not empty, do
  pick $\beta \in T_i$ randomly and set $T_i = T_i \setminus \{\beta\}$;
  for each unmarked member γ of Markov boundary of β, do
    if γ is observable, set $B_i = \{\gamma\} \cup B_i$;
    otherwise, do
      set $T_i = \{\gamma\} \cup T_i$ and set $Z_i = \{\gamma\} \cup Z_i$;
      mark γ;

Algorithm 4 (UnifyMarkovBoundary) *The system coordinator selects an agent A_r to run CollectMarkovBoundary in G_r. After it finishes, A_r runs DistributeMarkovBoundary in G_r. Finally, each agent A_i removes nodes it cannot observe from B_i;*

Algorithm 5 (CollectMarkovBoundary) *Denote all adjacent agents of an agent A_c by $A_{k1}, A_{k2}, ..., A_{kc}$. Agent A_c does the following:*

for each adjacent agent A_i ($i = k1, k2, ..., kc$) except caller, do
  call A_i to run CollectMarkovBoundary;
  A_c absorbs $B_i \cap V_c$ to B_c;

Algorithm 6 (DistributeMarkovBoundary) *Denote all adjacent agents of an agent A_c by $A_{k1}, A_{k2}, ..., A_{kc}$. Agent A_c does the following:*

for each adjacent agent A_i ($i = k1, k2, ..., kc$) except caller, do
  pass $B_c \cap V_i$ to B_i of A_i;
  call A_i to run DistributeMarkovBoundary;

By this way, the Markov boundary of a set of nodes can be computed across all subdomains through message passing over shared nodes only. This is a good property. Under the multiagent paradigm, each agent may be constructed by an independent vendor who embeds the know-hows about a subdomain into the agent. The vendor may not be willing to reveal the know-hows when the agent is integrated into a multiagent MSBN. To protect the know-hows of such vendors, it is desirable not to force each agent to reveal its internal structures.

We are going to demonstrate the computation of the observable Markov boundary of an unobservable node c in subnet G_1 shown in Figure 1. We assume nodes a, e, f, d and z are unobservable. We also assume A_0 is the host agent of $\{a, b, z\}$, A_1 is the host agent of $\{c, v, x\}$, A_2 is the host agent of $\{e, f, y\}$ and A_4 is the host agent of $\{d, u\}$. Since c is a private node, only $T_1 = \{c\}$ is not empty after initialization. Then agent A_1 runs ComputePartialMarkovBoundary

in G_1. The observable partial Markov boundary of c in G_1 would be $B_1 = \{x, u, v, y\}$. Node z is a Markov boundary of c. It is not in B_1 because it is unobservable. Hence, $Z_1 = \{z, c\}$. Node z is a node shared with agent A_0, so agent A_1 passes z to T_0 of A_0. Since A_1 does not have any unobservable nodes in Z_1 shared with agents A_2 and A_3, nothing is passed to A_2 and A_3. Since T_0 is not empty, restart ExpandMarkovBoundary. Agent A_0 runs ComputePartialMarkovBoundary in G_0. The observable partial Markov boundary of c in G_0 would be $B_0 = \{b, x\}$. Node a is unobservable, so it won't be in B_0. Therefore, $Z_0 = \{z, a\}$. A_0 passes z back to A_1. Since z was marked by A_1 already, T_1 becomes $\emptyset$ after A_1 removes marked nodes. Until now, T_0, T_1, T_2 and T_3 are all empty. UnifyMarkovBoundary is called to distribute all observable Markov boundary members to corresponding agents. The distribution can be started from any one agent. For example, we can start from agent A_0. Agent A_0 calls A_1 and A_1 calls A_2 and A_3 to run CollectMarkovBoundary. A_1 get nothing from A_2 and A_3. A_0 union $B_1 \cap V_0 = \{x\}$ to B_0. Since x is already in B_0, B_0 does not change. Then agent A_0 is called to run DistributeMarkovBoundary. It passes $B_0 \cap V_1 = \{x\}$ to B_1. Since x is in B_1, B_1 does not change. A_1 is then called to run DistributeMarkovBoundary. It passes $B_1 \cap V_2 = \{u, v\}$ to B_2 of A_2 and $B_1 \cap V_3 = \{x, y\}$ to B_3 of A_3. Finally, after each agent removes all nodes for which it is not host agent from the respective partial Markov boundary, the observable partial Markov boundaries of c in subnets G_0, G_1, G_2, and G_3 are respectively:

$$B_0 = \{b, x\} \setminus \{x\} = \{b\},$$
$$B_1 = \{u, v, x, y\} \setminus \{u, y\} = \{v, x\},$$
$$B_2 = \{x, y\} \setminus \{x\} = \{y\}, \text{and}$$
$$B_3 = \{u, v\} \setminus \{v\} = \{u\}.$$

Hence, over the MSBN, the Markov boundary of c is $B = B_0 \cup B_1 \cup B_2 \cup B_3 = \{b, u, v, x, y\}$.

Preliminary Experiments

Dynamic Domain

We shall demonstrate the approach on a sequential digital circuit simulator, which can construct a sequential digit circuit and simulate its running (An, Xiang, & Cercone 2003). The simulator supports multiagent inquires.

We use digital circuits as our test beds because they require little extra knowledge for the professionals in IT to understand. Perhaps this is one of the major reasons why digital electronics has been the source of problems for many researchers in diagnosis (Davis 1984; Genesereth 1984; de Kleer & Williams 1987; Poole 1993; Srinivas & Horvitz 1995; Xiang 1999; Xiang, Olesen, & Jensen 2000; Xiang 2002). A digital circuit system is intended to work deterministically, but the failure behavior is uncertain. Hence, the use of a digital domain does not diminish the number of general issues related with uncertain reasoning. Further more, the complexity in modeling and inference using probabilistic graphical models grows as the degree of network nodes increases and the number of loops in the network increases. In digital system, the former corresponds to the number of inputs and outputs for a particular gate or device, and the latter is reflected in the circuit topology. A digital system may be combinatorial or sequential. In a combinatorial circuit, output values depend only on the input values, whereas in a sequential circuit output values depend also on the internal state of the circuit which is determined by the history of inputs. Therefore, a combinatorial circuit system provides a static domain, whereas a sequential circuit system provides a dynamic domain.

Figure 4 shows a synchronous sequential digital circuit which is composed of two D flip-flop, 4 J-K flip-flops and 19 logic gates. Clock signal to 6 flip-flops is ignored in the diagram. The circuit is simulated by our simulator.

MSBN Model

We monitor and diagnose the simulated circuit with a 5 agent MSBN over a period of two time instants. To reason the state of a dynamic device, at least two time instants are needed; otherwise, we may not be able to know if a flip flop is faulty or not. Figure 5 shows the structure of the Bayesian subnet over subdomain C_2. The figure was generated using the WebWeavr toolkit. Each variable is labeled by its variable name followed by the variable's index (which readers may ignore) separated by a comma. The variable name is composed of a string indicating a device or a signal point in the digital circuit and a digit indicating time instant. For example, $d11$ denotes the state of J-K flip flop d_1 at relative time instant 1.

To use the MSBN to reason the state of the domain, we perform cooperative triangulation to get its linked junction forest (LJF)(Xiang 2001). The largest cliques in the linked junction forest of the MSBN are those of 6 nodes and the largest linkages in linkage trees (Xiang 2001) are those of 5 nodes. This indicates we can perform effective inference using the MSBN.

In addition to the dependence structure, we assume the following representational parameters. The state of a device (flip-flops, logic gates) at a time instant is represented by a binary variable and is either *normal* or *abnormal*. It is assumed that each device has a 0.01 probability of being faulty. A faulty device is modeled so that it may or may not produce the incorrect output. A faulty AND or Flip-Flop is assumed to output correctly 20% of the time. A faulty NOT is assumed to output correctly 50% of the time. A faulty OR gate is assumed to output correctly 70% of the time.

Monitoring

We assume J-K flip flop d_1 in subdomain C_2 is faulty. We assume the states of devices in digital circuits are unobservable. Hence, in Figure 5, both $d10$ and $d11$ are unobservable. To make observation efficient, we compute their Markov boundary by Algorithms 1, 2, 3, 4, 5, and 6 which contains 5 nodes: $\{s00, w01, w00, s01, u00\}$. Since s_0 and w_0 only exists in subdomain C_2, the host agent of $s00, s01, w00$ and $w01$ should be A_2. However, both agents A_2 and A_3 could be the host agent of entities $u00$ and $u01$ because u_0 exists in both subdomains C_2 and C_3. We assume u_0 physically exists in subdomain C_2 and hence agent A_2 is the host agent

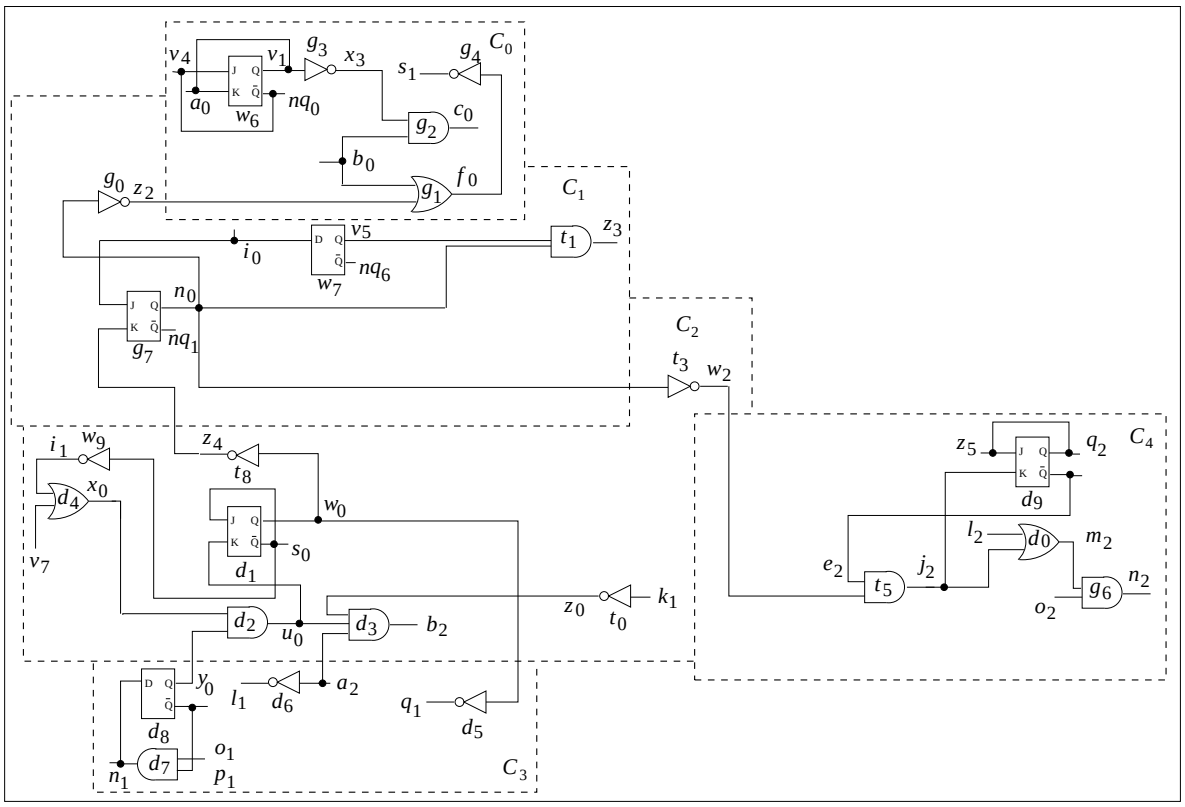

Figure 4: A sequential digital circuit.

of $u00$ and $u01$. So agent A_2 is the host agent of all these Markov boundary members and observes them. After that, all agents communicate with A_2 to absorb the impacts of the observation. Agent A_2 finds that d_1 is faulty (with belief 100%) and is sure nothing else is abnormal (with belief more than 98%). All other agents are very confident that nothing in their corresponding subdomains is abnormal (with belief more than 98%).

In the example above, we assume that the state of each device is unobservable. To make the situations more challenging, we may also assume that observation of each input and output has a cost. Hence, observing all inputs and outputs is not an option. We assume inputs and outputs s_0, w_0, u_0 of flip flop d_1 in Figure 4 are unobservable. On the assumption, the Markov boundary of $\{d10, d11\}$ becomes $\{a20, a21, b20, b21, i10, i11, x00, x01, y00, y01, z00, z01, z40, z41\}$. All these members exist in subdomain C_2, but they may not all have agent A_2 as their host agent. We assume A_3 is the host agent of $y00$, $y01$, $a20$ and $a21$ and A_2 is the host agent of the rest of Markov boundary members. After agents A_2 and A_3 observe these boundary members respectively in their own subdomains, all agents communicate with each other. Agent A_2, then, is quite confident that d_1 is faulty (with belief 97.08%) and believes nothing else in its subdomain is abnormal (with belief 94.11% for t_8 and belief 95.50% for w_9 and belief more than 98% for all other devices). All other agents believe that devices in their corresponding subdomains are normal (with belief more than 98%).

Discussion

This approach monitors the domain period by period. The problems of the domain will be reasoned about over and over again. The two consecutive periods could overlap on some instants. When the dynamic domain proceeds slow enough, reasoning with overlapped consecutive periods can find problems earlier. For example, the last period contains instants 3, 4 and 5 and the current period contains instants 4, 5, and 6. If problems happened at instant 5, the problems won't be found in the last period. This is because effects of any problems at instant 5 can only be reflected at and after instant 6. The earliest time to find the problems would be at instant 6. However, if the current period contains instants 6, 7, and 8 which does not overlap with the last period, the earliest time to find the problems would be at instant 7.

Markov boundary helps us locate most relevant variables to observe in graphical models. However, this won't guarantee we find problems. Due to the observabilities of nodes, observable Markov boundary may be far from the problem origins. It is possible we could never locate the exact problem origins if the environment we could observe is too far from the problem origins. On the other hand, Markov boundary may provide more than enough nodes to observe to find the problems. Sometimes, we could have had very high confidence that we identify the problem origins before observing all Markov boundary members.

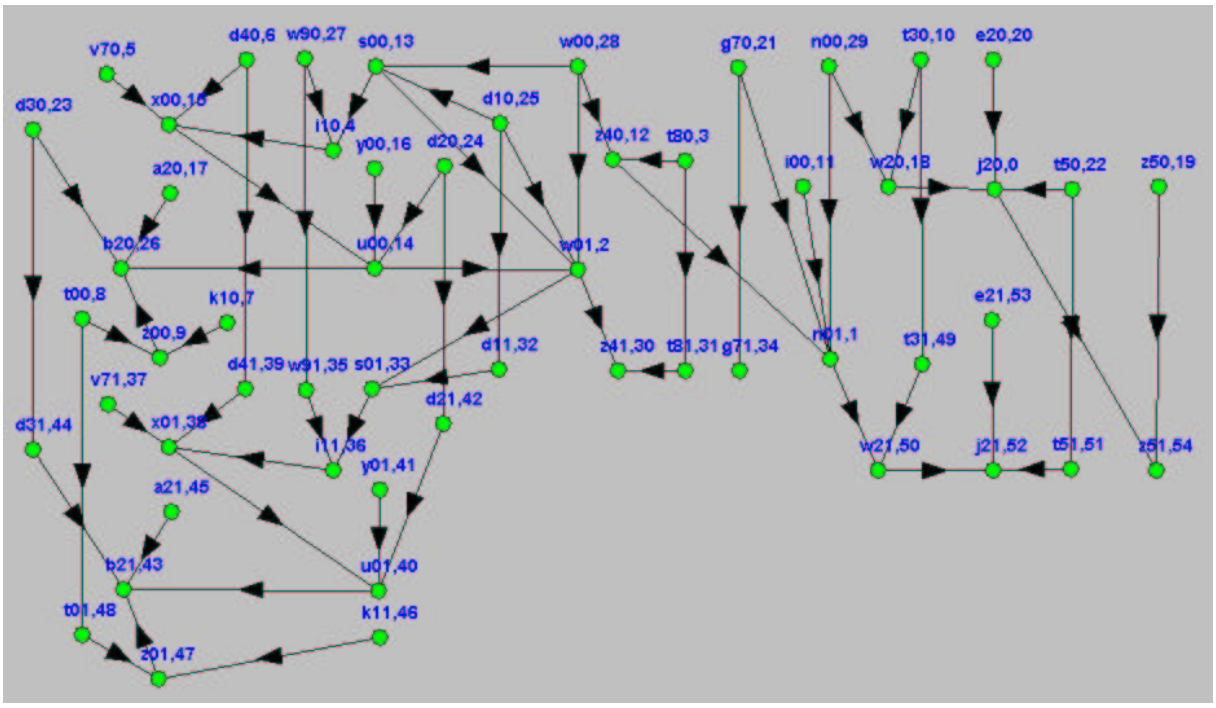

Figure 5: Subnet 2 (over subdomain C_2) of a two time instant MSBN for the digital circuit in Figure 4.

This approach reasons about the states of dynamic multiagent domains exactly in each period. Information from remote history is ignored, although information from the most recent history is absorbed. The approach would be most suitable to those domains (or cases) where history would affect the present softly. For example, in medical therapy domain, previous sickness information may not be very necessary (maybe helpful) for the therapy of the current sickness. The approach is proposed based on intuition that current state observation may play a more important role than remote history information (e.g. in medical therapy). Observation can also make some old evidence independent of the current state. New observation may correct belief revised from old unreliable observation. Further theoretical analysis or empirical study on this approach has to be made. For example, what kind of multiagent dynamic domains are most suitable to this approach and why? How to determine how many instants each period should contain to make inference both efficient and effective? We will also further test this approach on sequential digital circuits with multiple faults.

Conclusions

Two issues (distribution issue and decomposition issue) are involved in cooperative multiagent probabilistic reasoning in dynamic domains. Distribution issue shows that the spatial distribution of multiagent systems conflicts with the temporal dependencies of dynamic domains for probabilistic reasoning. Decomposition issue shows the decomposed representation of JPDs is very poorly supported in dynamic multiagent domains. These issues do not allow each agent to obtain historic information separately and benefit from each other's knowledge up to the relevant history. However, we believe that observation of the most recent state, instead of remote history, may play a more important role in the reasoning about the state of dynamic domains. We propose an approach to deal with these issues. This approach ignores effects of remote history and models a period of a dynamic multiagent domain into an MSBN. Then the state of the domain can be reasoned about period by period exactly. We compute and observe an observable Markov boundary of a set of suspected entities. This makes observation more efficient and relevant.

Preliminary experiments show the approach performs well when the observable Markov boundary of a set of suspected entities is close to the suspected entities.

Acknowledgements

This work is supported by the research grants to the second and the third authors from NSERC of Canada. We thank anonymous reviewers for their helpful comments.

References

An, X.; Xiang, Y.; and Cercone, N. 2003. A dynamic environment simulator. In *Proceedings of European Simulation and Modeling Conference*, 187–192. Naples, Italy: The European Simulation Society.

An, X. 2003. Probabilistic reasoning in dynamic domains under the cooperative multiagent paradigm. Technical report, School of Computer Science, University of Waterloo, Ontario, Canada.

Bilmes, J. A., and Bartels, C. 2003. On triangulating dynamic graphical models. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-2003)*. Acapulco, Mexico: Morgan Kaufmann Publishers.

Boutilier, C.; Friedman, N.; and Halpern, J. Y. 1998. Belief revision with unreliable observations. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-1998)*, 127–134. Madison, USA: AAAI Press.

Boyen, X., and Koller, D. 1998. Tractable inference for complex stochastic processes. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-1998)*, 33–42. San Francisco, CA: Morgan Kaufmann Publishers.

Castillo, E.; Gutierrez, J. M.; and Hadi, A. S. 1997. *Expert Systems and Probabilistic Network Models*. Springer.

Crick, C., and Pfeffer, A. 2003. Loopy belief propagation as a basis for communication in sensor networks. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-2003)*, 159–166. Acapulco, Mexico: Morgan Kaufmann Publishers.

D'Ambrosio, B. 1999. Inference in Bayesian networks. *AI Magazine* 20(2):21–36.

Darwiche, A. 2001. Constant space reasoning in dynamic Bayesian networks. *International Journal of Approximate Reasoning* 26:161–178.

Davis, R. 1984. Diagnostic reasoning based on structure and behavior. *Artificial Intelligence* 24:347–410.

de Campos, L. M., and Fernandez-Luna, J. M. 2002. Reducing propagation effort in large polytrees: an application to information retrieval. In *Proceedings of the First European Workshop on Probabilistic Graphical Models (PGM-2002)*, 35–44.

de Kleer, J., and Williams, B. C. 1987. Diagnosing multiple faults. *Artificial Intelligence* 32:97–130.

Dean, T., and Kanazawa, K. 1988. Probabilistic temporal reasoning. In *Proceedings of the 7th National Conference on Artificial Intelligence (AAAI-88)*, 524–528. St. Paul, Minnesota: AAAI Press.

Dean, T., and Kanazawa, K. 1989. A model for reasoning about persistence and causation. *Computational Intelligence* 5(3):142–150.

Friedman, N., and Halpern, J. Y. 1996. A qualitative Markov assumption and its implications for belief change. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence (UAI-96)*. Morgan Kaufmann Publishers.

Genesereth, M. R. 1984. The use of design description in automated diagnosis. *Artificial Intelligence* 24:411–436.

Kjaerulff, U. 1992. A computational scheme for reasoning in dynamic probabilistic networks. In *Proceedings of Eighth Conference on Uncertainty in Artificial Intelligence (UAI-92)*, 121–129. San Mateo, CA: Morgan Kaufmann Publishers.

Koller, D., and Pfeffer, A. 1997. Object-oriented Bayesian networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-1997)*, 302–313. San Francisco, CA: Morgan Kaufmann Publishers.

Lauritzen, S. L., and Spiegelhalter, D. J. 1988. Local computation with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society* B 50(2):157–224.

Murphy, K.; Weiss, Y.; and M.I.Jordan. 1999. Loopy belief propagation for approximate inference: an empirical study. In *Proceedings of fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-99)*, 467–475. Stockholm, Sweden: Morgan Kaufmann Publishers.

Murphy, K. 2002. *Dynamic Bayesian Networks: Representation, Inference and Learning*. Ph.D. Dissertation, CS Division, UC Berkeley, Berkeley, California.

Pearl, J. 1986. Fusion, propagation, and structuring in belief networks. *Artificial Intelligence* 29:241–288.

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

Poole, D. 1993. Probabilistic horn abduction and Bayesian networks. *Artificial Intelligence* 64:81–129.

Srinivas, S., and Horvitz, E. 1995. Exploiting system hierarchy to compute repair plans in probabilistic model-based diagnosis. In *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence(UAI-95)*, 523–531. Montreal, Canada: Morgan Kaufmann Publishers.

Xiang, Y., and Lesser, V. 2000. Justifying multiply sectioned Bayesian networks. In *Proceedings of the Sixth International Conference on Multi-agent Systems*, 349–356.

Xiang, Y.; Olesen, K. G.; and Jensen, F. V. 2000. Practical issues in modeling large diagnostic systems with multiply sectioned Bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence* 14(1):59–71.

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

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

Xiang, Y. 1999. Temporally invariant junction tree for inference in dynamic Bayesian networks. In *Artificial Intelligence Today: Recent Trends and Developments*, 473–489. Springer-Verlag.

Xiang, Y. 2001. Cooperative triangulation in msbns without revealing subnet structures. *International Journal of Approximate Reasoning* 37(1):53–65.

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

Zhang, H.; Tian, F.; and Lu, Y. 2001. A general algorithm for approximate inference in multiply sectioned Bayesian networks. In *Proceedings of Fourth International Symposium on Intelligent Data and Analysis*, 330–339. Lisbon, Portugal: Springer.