Towards More Efficient Inference in Dynamic Bayesian Networks

Yang Xiang

Technical Report CS-96-04
March, 1996

Department of Computer Science
University of Regina
Regina, Saskatchewan, CANADA
S4S 0A2

ISSN 0828-3494
ISBN 0-7731-0318-X

Towards More Efficient Inference in Dynamic Bayesian Networks

Y. Xiang

Department of Computer Science
University of Regina
Regina, Saskatchewan, Canada S4S 0A2
E-mail: yxiang@cs.uregina.ca

March 14, 1996

Abstract

The constrained node elimination (CNE) method is a method explicitly designed for *exact* inference in dynamic Bayesian networks (DBNs). It is sufficiently effective for applications where the dynamic domains evolve relatively slowly, but is reported to be "too expensive" for some fast evolving domain, e.g., the BATmobile, where the inference computation is under time pressure.

This research applies the framework of multiply sectioned Bayesian networks (MSBNs) to exact inference in stationary DBNs to improve the efficiency of the inference computation. We show how to create a temporally invariant representation (called a template junction tree) of a DBN directly capable of exact inference. This removes the need for the computation associated with dynamic expansion and reduction of the CNE method. We also show how proper sectioning of a DBN into a MSBN may cut down the size of the total state space of the template junction tree and hence improve the computational efficiency. Finally, we show how conditional independencies along the 'ortho-temporal' direction, ignored by the CNE method, can be explored to further cut down the total state space when the DBN is 'slim'.

`keywords`: probabilistic reasoning, knowledge representation, temporal reasoning, real-time systems.

`Acknowledgement`: This work is supported by the Research Grant OGP0155425 from NSERC.

1 Introduction

A dynamic Bayesian network (DBN) [3, 9] is an extension of Bayesian networks [12, 11, 10, 7] from static domains to dynamic domains, i.e., domains that change their states with time. A DBN consists of a finite number of 'slices' each of which is a domain dependence model at a particular time interval. Slices corresponding to successive intervals are connected through arcs that represent how the state of the domain evolves with time. Collectively, the slices represent the dynamic domain over a period of time.

When inference must be performed over an extended period of time, it is not feasible to maintain all slices accumulated in the past. Kjaerulff [9] proposed a method, which we shall refer to as the *constrained node elimination* (CNE) method, to perform *exact* probabilistic inference by dynamically adding new slices and cutting off old slices. It is the first method [9] and so far, to the best of our knowledge, still is the only known method explicitly designed for exact inference in DBNs. However, as the networks become more complex, the CNE method does not provide satisfactory performance in time-critical applications as reported by the BATmobile researchers [5]. They have instead turned to approximate inference method [8] in order to improve the performance.

In this paper, we apply the framework of multiply sectioned Bayesian networks (MSBNs) [15] to exact inference in DBNs. We show that by using the MSBN framework, significant improvement of the run-time performance over the CNE method may be obtained. We identify the conditions under which the improvement is possible.

The rest of the paper is organized as follows: Section 2 introduces DBNs. Section 3 briefly reviews the CNE method and Section 4 outlines the possible improvement over the CNE method. Section 5 briefly introduces the MSBN framework. Section 6, 7 and 8 show how to apply MSBNs to improve the efficiency of inference in DBNs. Section 9 draws some conclusions from this research.

2 Dynamic Bayesian Networks

We shall assume that readers are familiar with commonly used graphtheoretic terms like 'links' in undirected graphs, 'arcs' in directed graphs, 'head' and 'tail' of an arc, 'paths', 'cycles', 'triangulated graph', 'cliques', 'junction trees', etc. A DBN [3, 9] is a quadruplet

$$\mathcal{G}^K = (\bigcup_{i=0}^{K} N_i, \bigcup_{i=0}^{K} E_i, \bigcup_{i=1}^{K} F_i, \bigcup_{i=0}^{K} P_i).$$

Each N_i is a set of nodes labelled by variables. N_i represents the state of a dynamic domain at a particular time interval indexed by i $(i = 0, \ldots, K)$. Collectively, $\bigcup_{i=0}^{K} N_i$ represents the states of the dynamic domain over $K+1$ consecutive intervals. Each E_i is a set of arcs between nodes in N_i, which represent conditional independencies between domain variables at a given interval. Each F_i is a set of *temporal* arcs each of which is

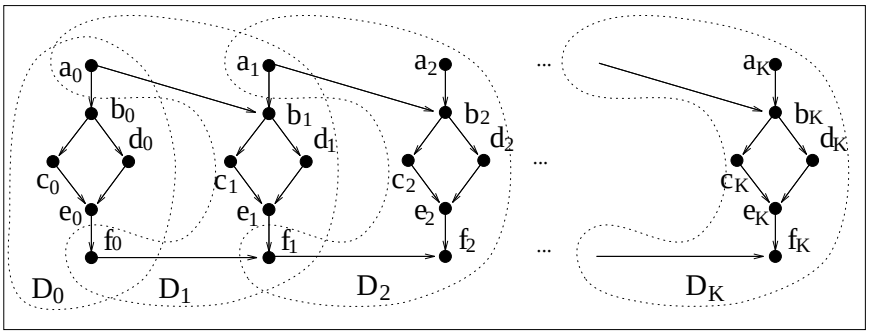

Figure 1: A dynamic Bayesian network.

directed from a node in N_{i-1} to a node in N_i $(i = 1, \ldots, K)$. These arcs represent the Markov assumption: the future states of the domain is conditionally independent of the past states given the present state. The subset of N_i $(0 \leq i < K)$

$$FI_i = \{x \in N_i | (x, y) \in F_{i+1}\}$$

is called the *forward interface* of N_i, where (x, y) is a temporal arc from x to y. The subset of N_i $(0 < i \leq K)$

$$BI_i = \{y \in N_i | (x, y) \in F_i\} \cup \{z \in N_i | z \in \pi(y) \ \& \ (x, y) \in F_i\}$$

is called the *backward interface* of N_i, where $\pi(y)$ is the set of parent nodes of y. Arcs of E_i and F_i are so directed such that $D_i = (N_i \cup FI_{i-1}, E_i \cup F_i)$ is a directed acyclic graph (DAG). Each P_i is a set of probability tables one for each variable x in N_i conditioned on $\pi(x)$.[1] The pair $S_i = (D_i, P_i)$ is called a *slice* of the DBN and D_i is called the structure of S_i. Collectively, the slices of a DBN define a Bayesian network, whose structure is the union of slice structures and whose joint probability distribution (jpd) is the product of probability tables in all slices.

Figure 1 shows the structure of a DBN where $N_1 = \{a_1, b_1, c_1, d_1, e_1, f_1\}$, $E_1 = \{(a_1, b_1), (b_1, c_1), (b_1, d_1), (c_1, e_1), (d_1, e_1), (e_1, f_1)\}$, $F_1 = \{(a_0, b_1), (f_0, f_1)\}$, $FI_1 = \{a_1, f_1\}$ and $BI_1 = \{a_1, b_1, e_1, f_1\}$.

At any current time $j \leq K$, the slices $S_0, \ldots, S_{j-1}$ represent the domain history and $S_{j+1}, \ldots, S_K$ predict the future. Evidence (observations obtained in the past and present) may be entered into $S_0, \ldots, S_j$. Limited by computational resource, normally only $S_i, \ldots, S_K$ $(i \leq j \leq K)$ are explicitly maintained, called *active slices* of the DBN. As time j increases, one or more new slices will be added on the front (called *expansion*) and some or all slices in the history will be cut off (called *reduction*) after the evidence contained in them have been propagated to the remaining active slices.

[1] The tables for variables in FI_{i-1} are not explicitly specified but determined by $P_0, \ldots, P_{i-1}$.

We assume that the dynamic domain is *stationary*, i.e., the structure as well as the conditional probabilities of each slice S_i does not change with time i. We shall say that the corresponding DBN is stationary.

3 The Constrained Node Elimination Method

This section briefly reviews the CNE method [9]. Section 3.1 covers expansion and Section 3.2 covers reduction.

A node x in an undirected graph $G = (N, E)$ is *eliminated* if the adjacency $adj(x)$ of x is made complete (each pair of nodes is adjacent) by adding necessary links (if any) before x and links incident to x are removed. Each link added during the elimination is called a *fill-in*. Let α be an arbitrary order of nodes in G and ρ be the set of fill-ins added in the process of eliminating all nodes of G in the order α. Then the graph $G' = (N, E \cup \rho)$ is triangulated.

3.1 Expansion of active slices

We use an example DBN from [9] to illustrate the method. Only Figure 2 (a) and its moral graph were given in the reference without the details for other steps. We illustrate the method in more details here for later comparison with our method.

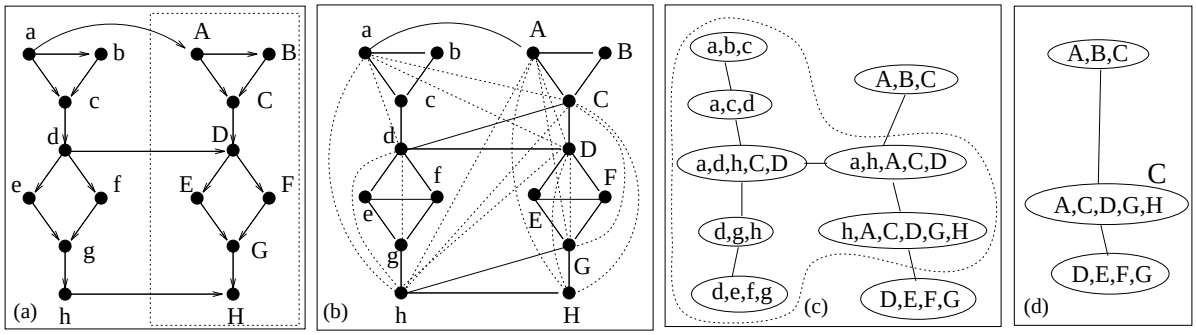

Figure 2: Expansion and reduction in DBN using the CNE method. (a) A new slice is added to the single active slice. (b) Expanded slices are moralized and triangulated by constrained node elimination. (c) The JT after expansion. (d) The JT after reduction.

- The slices corresponding to future time are added at the front of the active slices of the DBN. In Figure 2 (a), a single active slice at time i is assumed and its structure is drawn in the left. The temporal arcs into the slice are not shown. The added structure for time $i + 1$ is shown in the dotted box.

- Moralization is performed on the new active slices, i.e., parents of each node is made complete and directions of arcs are dropped. The resultant *moral graph* of Figure 2 (a) is shown in (b) with solid links.

- Triangulation is performed on the resultant graph by node elimination in a *constrained* elimination order. According to a constrained order, a node in N_i must be eliminated before a node in N_{i+1}. This constrained triangulation has the property that the backward interface BI_{i+1} is made complete and therefore is contained in a single clique. The triangulated graph for Figure 2 (a) is shown in (b) and the fill-ins are shown as dotted links. It is obtained using the constrained elimination order

$$b, e, f, c, g, d, a, h, B, E, F, C, G, D, A, H$$

  as suggested in [9]. Note that $BI_{i+1} = \{A, C, D, G, H\}$ is complete.

- Cliques of the triangulated graph are identified and a junction tree (JT) of cliques are constructed. The JT of Figure 2 (b) is shown in (c).

- Belief tables (unnormalized probability tables) are associated with cliques and belief propagation is performed such that the evidence accumulated in the history is absorbed into the active slices. The expansion is now completed.

3.2 Reduction of active slices

Let T be the JT of the active slices. Let $\mathcal{C}$ be the set of cliques in T. Let $\mathcal{C}'$ be the subset of cliques in T that contain variables from the slices to be removed. Let $\mathcal{C}'' = \mathcal{C} \setminus \mathcal{C}'$ be the set of remaining cliques. Let the subset of cliques in $\mathcal{C}''$ that are adjacent to cliques in $\mathcal{C}'$ be $\mathcal{B} = \{C \in \mathcal{C}'' | adj(C) \cap \mathcal{C}' \neq \phi \text{ in } T\}$. Suppose all slices up to S_i *inclusive* are to be removed. After the expansion described above, the reduction proceeds as follows:

- If there is no clique $C \in \mathcal{B}$ such that $BI_{i+1} \subseteq C$ where BI_{i+1} is the backward interface of slice S_{i+1}, then add a clique $C = BI_{i+1}$ to $\mathcal{C}''$ with $adj(C) = \mathcal{B}$. Otherwise, let $adj(C) = \mathcal{B} \setminus \{C\}$.

- Associate C with the proper belief table. The reduction is now completed.

In Figure 2 (c), $\mathcal{C}'$ is the set of cliques enclosed in the dotted line. $\mathcal{C}''$ and $\mathcal{B}$ are identical in this case. After cliques in $\mathcal{C}'$ are removed and the clique $C = BI_{i+1}$ is added, the reduced JT is shown in (d).

4 Improvement Over The CNE Method

The CNE method is sufficiently effective for applications where the dynamic domains evolve relatively slowly, e.g., monitoring the effect of medical therapy [1] or commercial forecasting [2]. However, for fast evolving domains where inference computation is under time pressure, e.g., mobile robot navigation [4] or automated vehicles [5], the method appears to be "too expensive" as reported by the BATmobile researchers [5].

We attribute the unsatisfactory performance of the CNE method to its following components:

1. During expansion and reduction, an updated representation, a new JT for the modified active slices (with the associated potentials) is constructed. The construction involves expensive computation. When the domain is stationary, the slices of the DBN are invariant with time. If the number of active slices is a constant, then it is possible to create a representation for inference that is invariant with time.

   The BATmobile group explored along this direction [5]. They proposed to precompile the slice of a stationary DBN into a 'temporally invariant network' for which there exists a node elimination order which induces the same slice structure as the original. However, since the approach that they took was to replace exact inference by approximate inference using stochastic simulation, they did not deal with the issue of establishing a stable representation directly capable of *exact* inference. In Section 6, we address this issue using the framework of MSBNs.

2. The CNE method has the property that the backward interface BI_i is contained in a single clique C. As the cardinality of BI_i increases, the size of C increases. This results in the increase of the size of the total state space of the new JT, which in turn causes the increase of the complexity of expansion, reduction as well as inference computation. In Section 7 and 8, we show how to apply MSBNs to construct a stable data structure that has a much smaller total state space than that obtained by the CNE method.

5 Multiply Sectioned Bayesian Networks

This section briefly introduces MSBNs. More details can be found in [15]. A MSBN M consists of a set of interrelated Bayesian subnets. Each subnet represents dependencies of a subdomain in a large problem domain or total universe. Each subnet shares a nonempty set of variables with at least one other subnet. The intersection between each pair of subnets forms a *d-sepset* as defined below.

Definition 1 (d-sepset) *Let* $D^i = (N^i, E^i)$ $(i = 1, 2)$ *be two DAGs such that* $D = (N^1 \cup N^2, E^1 \cup E^2)$ *is a DAG. The intersection* $I = N^1 \cap N^2$ *is a* `d-sepset` *between* D^1 *and* D^2 *if, for every* $A_i \in I$ *with its parents* π_i *in* D*, either* $\pi_i \subseteq N^1$ *or* $\pi_i \subseteq N^2$*.*

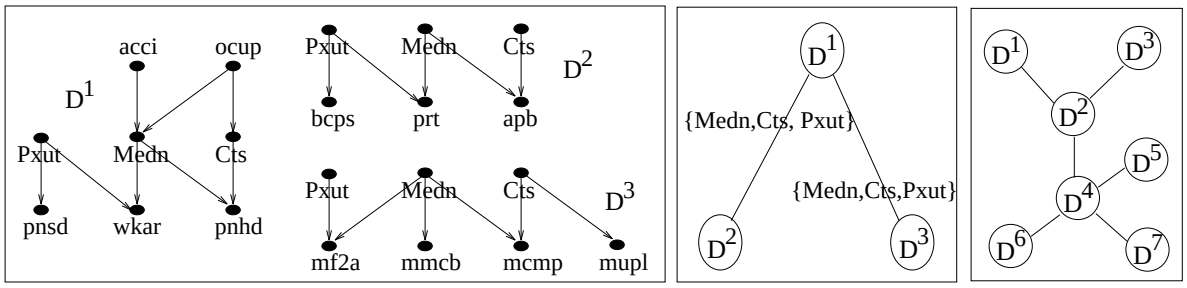

Figure 3: Left: An example MSBN for neural muscular diagnosis. Middle: The hypertree organization of the MSBN in the left. Right: A general hypertree structured MSBN.

It can be shown that, when a pair of subnets are isolated from M, their d-sepset renders them conditionally independent. Figure 3 (left) shows the structure of a MSBN for diagnosis of Median nerve lesion (Medn), Carpal tunnel syndrome (Cts) and Plexus upper trunk lesion (Plut).[2] It consists of three subnets D^i $(i = 1, 2, 3)$ for clinical, electromyography and nerve conduction subdomains, respectively. The d-sepset between each pair of subnets is $\{Medn, Cts, Plut\}$. In general, d-sepsets between different pairs of subnets of M may be different.

Subnets of M are organized into a *hypertree* structure. Each hypernode is a subnet of M. Each hyperlink is a d-sepset between a pair of subnets. A hypertree structured M ensures that each hyperlink render the two parts of M that it connects conditionally independent. The subnets in Figure 3 (left) can be organized into the hypertree in Figure 3 (middle). Figure 3 (right) depicts a general hypertree structured MSBN.

Each subnet in M may be multiply connected (more than one path between a pair of nodes), e.g., D^1. In order to perform inference more efficiently in each subnet, the hypertree structured M is converted into a *linked junction forest* (LJF) F of the identical hypertree structure as its run time representation. Each hypernode in the hypertree is a JT converted from the corresponding subnet through moralization and triangulation. Each hyperlink in the hypertree is a set of *linkages* which covers the d-sepset between the two corresponding subnets.

The need for linkages can be understood as follows: When evidence is obtained in one subnet/JT, it can be propagated to an adjacent JT by passing the probability distribution over the d-sepset I. This may not be efficient if the cardinality of I is large. The efficiency can be improved by exploiting the conditional independence within I. Linkages form a decomposition of I based on conditional independence. Once linkages are defined, the probability distribution over I can be passed by passing distributions over linkages, which is more efficient. We will illustrate this later in this section. Linkages are obtained as follows:

[2] The example is taken from a fraction of PAINULIM [14] with modification.

Definition 2 (linkage) *Let I be the d-sepset between adjacent JTs T^a and T^b in a LJF.*

Remove recursively every leaf clique C of T^a that satisfies one of the following conditions. (1) $C \cap I = \phi$. (2) $C \cap I$ is a subset of another clique. Denote the resultant graph by T'.

Then remove recursively either a node from a clique of T' or a clique from T' as follows. (a) If a node $x \notin I$ is contained in a single clique C, remove x from C. (b) If a node $x \notin I$ is contained in several cliques, union the cliques into one clique C and remove x from C. (c) If a clique C becomes a subset of an adjacent clique D after (a) or (b), union C into D.

The resultant is a `linkage tree` *$Y^{a \to b}$ of T^a relative to T^b. Each clique l of $Y^{a \to b}$ is a* `linkage` *from T^a to T^b. The clique of T^a that contains a linkage l is the* `linkage host` *of l.*

It can be shown that a linkage tree is a JT. It can also be shown that belief propagation between JTs through linkages can be performed correctly if and only if $Y^{a \to b}$ and $Y^{b \to a}$ are identical.

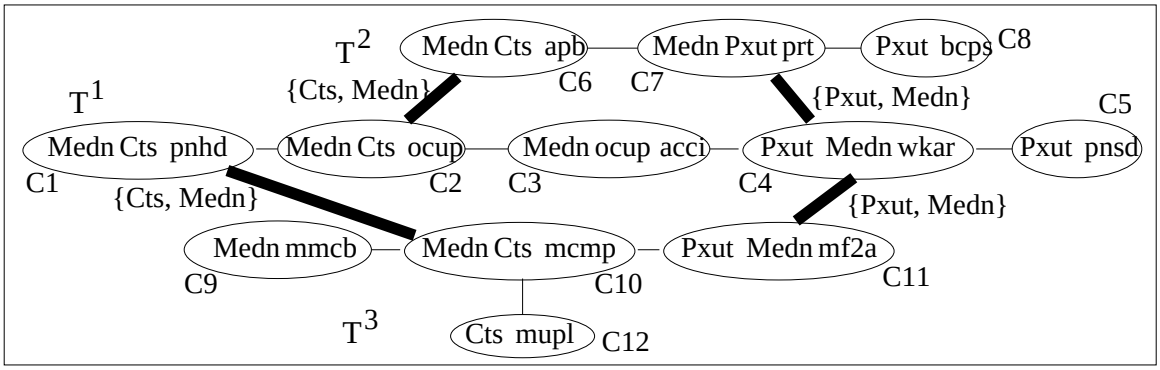

Figure 4: A linked junction forest of the MSBN in Figure 3.

The MSBN in Figure 3 (left and middle) can be converted into the LJF in Figure 4. The three subnets D^i $(i = 1, 2, 3)$ are converted into three JTs T^i $(i = 1, 2, 3)$. Then linkages (shown as heavy links) between pairs of JTs are defined. The linkage tree of T^2 relative to T^1 is obtained by first removing the clique $C8$, and then removing the variable *apb* from the clique $C6$ and removing *prt* from $C7$. We then obtained the linkage tree with two cliques $\{Cts, Medn\}$ and $\{Pxut, Medn\}$ each of which is a linkage between T^1 and T^2. Their linkage hosts in T^2 are $C6$ and $C7$, and their hosts in T^1 are $C2$ and $C4$.

Parallel to the structure conversion, the conditional probability tables stored at nodes of M are converted to belief tables of cliques in JTs of F such that a *joint system belief* of F, assembled from the belief tables, is equivalent to the jpd of M. The belief table of

a JT T is

$$B_T(N) = \prod_i B_{C_i}(C_i) / \prod_j B_{S_j}(S_j), \tag{1}$$

where N is the set of domain variables of T, $B_{C_i}(C_i)$ is the belief table of clique C_i and $B_{S_j}(S_j)$ is the belief table of clique separator S_j. Subscripts are used to denote the object that a belief table is associated with. Let $B_I(I)$ be the belief table of a d-sepset I assembled from belief tables of linkages in the corresponding linkage tree in the similar fashion as Equation 1 (recall that a linkage tree is a JT). The joint system belief of F takes the form

$$B_F(U) = \prod_i B_{T^i}(N^i) / \prod_j B_{I^j}(I^j), \tag{2}$$

where $U = \cup_i N^i$ is the total universe. Since belief tables are unnormalized probability distributions, $B_F(U)$ is proportional to the jpd of F

$$P_F(U) = \prod_i P_{T^i}(N^i) / \prod_j P_{I^j}(I^j), \tag{3}$$

where P denotes a probability distribution.

To answer queries by efficient local computation in F, it must be consistent. F is *locally consistent* if all JTs are internally consistent, i.e., when marginalized onto the same set of variables, different belief tables in a JT yield the identical marginal distribution. F is *boundary consistent* if each pair of adjacent JTs are consistent with respect to their d-sepset. F is *globally consistent* if it is both locally consistent and boundary consistent.

A set of operations are developed to achieve consistency during evidential reasoning: We assume that F is initially globally consistent. Details on initialization can be found in the above references.

After evidence is entered into a JT, the JT is no longer internally consistent and F is no longer globally consistent. `UnifyBelief` brings a JT internally consistent. It is defined in terms of `DistributeEvidence` (an outward belief propagation within a JT) and `CollectEvidence` (an inward belief propagation) proposed by Jensen et al [7].

Operation 3 (UnifyBelief) *Let T be a JT in a LJF. When* `UnifyBelief` *is initiated in T, the following are performed: (1) A clique C is arbitrarily selected. (2)* `CollectEvidence` *is called in C. (3) When C has finished* `CollectEvidence`, `DistributeEvidence` *is called in C.*

For example, suppose `UnifyBelief` is performed in T^3 of Figure 4 and the clique $C9$ is selected. During `CollectEvidence`, first belief propagates from $C11$ to $C10$ and from $C12$ to $C10$, and then belief propagates from $C10$ to $C9$. During `DistributeEvidence`, first belief propagates from $C9$ to $C10$, and then from $C10$ to $C11$ and $C12$. This brings T^3 internally consistent.

When evidence is available relative to variables in a JT, it is entered by `EnterEvidence`. `EnterEvidence` enters evidence by multiplying the belief tables of relevant cliques with the evidence function and then brings the JT internally consistent again by calling `UnifyBelief`.

For example, suppose Median motor conduction block ($mmcb = true$) is observed in the nerve conduction study of a patient. In Figure 4, the clique $C9$ contains the variable $mmcb$. During `EnterEvidence`, the belief table of $C9$ will be modified such that all configurations of $\{Medn, mmcb\}$ incompatible with the observation will be set to 0. Then `UnifyBelief` is called in $C9$.

Belief propagation between adjacent JTs in F are performed with `UpdateBelief`. It updates the belief of a JT T relative to an adjacent JT, and brings T internally consistent. It is defined in terms of a lower level operation `AbsorbThroughLinkage`. Given a linkage and its two hosts (one at each JT involved), `AbsorbThroughLinkage` updates the belief table of one host by the marginalization of the belief table of the other host over the linkage.

Operation 4 (UpdateBelief) [3] *Let $L = \{L_1, \ldots, L_k\}$ be the set of linkages between JTs T^a and T^b. Let C_i^a and C_i^b be the linkage hosts of L_i in T^a and T^b, respectively. When* `UpdateBelief` *is called in T^a to update its belief relative to T^b, the following are performed:* `AbsorbThroughLinkage` *is called in each C_i^a to absorb from C_i^b through L_i. After each* `AbsorbThroughLinkage`, `DistributeEvidence` *is called in C_i^a.*

In Figure 4, suppose `UpdateBelief` is called in T^2 to update its belief relative to T^1. First, belief propagates from $C2$ to $C6$ through the linkage $\{Cts, Medn\}$ followed by `DistributeEvidence` in $C6$. Then belief propagates from $C4$ to $C7$ through the linkage $\{Pxut, Medn\}$ followed by `DistributeEvidence` in $C7$. Note that if all variables in the d-sepset $\{Medn, Cts, Pxut\}$ has three possible values, then the belief table over the d-sepset has 27 values. Exploring conditional independencies within d-sepset, we only pass belief tables over linkages with 9+9=18 values.

Global consistency during evidential reasoning is maintained by `ShiftAttention`. After the user has entered multiple pieces of evidence into a JT, `ShiftAttention` allows the user to shift attention to another target JT. It maintains consistency along the hyperpath in the hypertree from the current JT to the target JT.

Operation 5 (ShiftAttention) *Let $T^0, T^1, \ldots, T^j$ be a subset of JTs in a LJF that form a simple path in the hypertree from T^0 to T^j. When* `ShiftAttention` *is called to shift attention from T^0 to T^j, for $i = 1$ to j,* `UpdateBelief` *is called in T^i to update its belief relative to T^{i-1}.*

[3] A more efficient version of `UpdateBelief` can be found in [13].

In Figure 3 (right), suppose the user currently focuses his attention on D^1/T^1.[4] If he wants to shift attention to D^3, `ShiftAttention` will propagate belief from D^1 to D^2 and then to D^3. Note that subnets $D^4, \ldots, D^7$ are not computed during this `ShiftAttention`.

A user may start with a particular JT, enter some evidence, query the subnet, shift attention to another JT, and repeat these actions for a number of times. It can be shown that, after `ShiftAttention`, the target JT is always consistent at the *global level* in the sense that answers to queries provided by the JT is consistent with all the evidence entered so far in the entire LJF.

6 MSBN as Temporally Invariant Representation

In this section, we first show *conceptually* how a DBN can be represented as a MSBN and how inference can be performed using the MSBN operations. We then discuss how we can use the MSBN framework to gain *computational* efficiency.

We assume that the number of active slices of the DBN is a constant $m \geq 1$. This will be relaxed later. Consider a DBN of k m ($k \geq 1$) slices representing the states of a dynamic system from the initial time 0 to some future time k $m-1$. Using the MSBN framework, it can be *sectioned* into a MSBN with k subnets, each of which contains m slices and represents a temporal subdomain over a period from i to $i+m-1$ for some i. By the stationary assumption, all slices are identical and hence all subnets of the MSBN are identical. Since each subnet is adjacent to at most two other subnets, one in the immediate history and another in the immediate future, the hypertree structure for this MSBN becomes a hyperchain. This MSBN is transformed into a LJF consisting of identical JTs.

Evidential reasoning in the DBN corresponds to a sequence of attention shifts in the LJF from one end of the hyperchain to the other end. Between time 0 and $m-1$, variables corresponding to evidence are contained in the first JT of the hyperchain. This is equivalent to when user's attention is focused on the corresponding temporal subdomain. Evidence can be entered into this JT using `EnterEvidence` and posterior probabilities can then be computed locally. At time m, variables corresponding to evidence are no longer contained in the first JT. This is equivalent to when user's attention is shifted to the next temporal subdomain. The operation `ShiftAttention` can then be applied to propagate evidence acquired between time 0 and $m-1$ to the second JT. Evidence available between time m and $2m-1$ can now be entered and the inference continues in this fashion.

Unlike inference in a general MSBN where user may enter evidence to a JT visited before, normally no evidence will be entered to any JT in the history. Therefore, no JT visited before needs to be maintained. Since every JT in the LJF is the same, no JT in the future needs to be maintained either. In summary, only a single JT, corresponding to the

[4] The diagram in fact shows the MSBN M, not the LJF F. We abuse the illustration a bit since M and F share the same hypertree structure.

active slices, is to be maintained at any time and `ShiftAttention` works in a particularly efficient form *computationally* as follows.

Let T be the identical JT which can be stored as a template. Let T' be the active JT (resident in the memory) which is identical to the template except that it contains evidence entered and relevant information about the history. Let T'' be the next JT following T' in the hyperchain. In order to shift attention from T' to T'', we first cache belief tables on linkages between T' to T''. We then copy belief tables of T into those of T', which effectively turns T' into T. We perform `UpdateBelief` in T relative to T' using the cached belief tables, which updates T into T''. Now T'' contains all relevant information about the history and is ready to receive new evidence.

We emphasize that the above inference involves only a single (stable) representation, i.e., the template JT. Most of the computation required during expansion and reduction according to the CNE method, e.g., triangulation, identification of cliques, creation of the JT and creation of belief tables are no longer needed. We illustrate this in Section 7.

7 Forward Interface Sectioning to Reduce State Space

Recall from Section 3 that the CNE method has the property that the backward interface BI_i must be complete. This property can be viewed as a constraint (a subset X of nodes must be complete) on triangulation. In general, such a constraint tends to increase the total state space of the JT resultant from the triangulated graph through two mechanisms: (**M1**) At least one clique that contains X and therefore has a cardinality $\geq |X|$ will be formed. Without this constraint, nodes in X may be distributed in several smaller cliques. (**M2**) When X is forced complete, fill-ins are added between non-adjacent nodes in X and they may not be needed in a triangulation without the constraint. The fill-ins may introduce untriangulated cycles which in turn cause more links to be added in order to triangulate the graph. These links increase the size of cliques resulting from the triangulation.

Due to **M1** and **M2**, the size of total state space (STSS) of the JT constructed by the CNE method increases rapidly as the cardinality of BI_i increases. Since the complexity of the inference computation is directly affected by STSS, the computational expense increases accordingly. In this section we explore the possibility of cutting down STSS by not requiring completion of BI_i.

Under the MSBN framework, the interface between two subnets is required to be a d-sepset. It can be easily shown that BI_i is a d-sepset. Therefore, the d-sepset interface is more general and therefore more flexible than the backward interface implied by the CNE method. Using the d-sepset criterion, we will show that the forward interface FI_i generally produces a JT with a smaller STSS. First, we show that FI_i is a d-sepset and hence is a qualified interface.

Proposition 6 *Let the structures of two adjacent subnets be* Π *and* Φ *defined as follows:*

$$\Pi = ((\bigcup_{i=j-m+1}^{j} N_i) \bigcup FI_{j-m}, \bigcup_{i=j-m+1}^{j} (E_i \cup F_i))$$

$$\Phi = ((\bigcup_{i=j+1}^{j+m} N_i) \bigcup FI_j, \bigcup_{i=j+1}^{j+m} (E_i \cup F_i)),$$

i.e., Π *is the composite structure of slices* $S_{j-m+1}, \ldots, S_j$ *and* Φ *is the composite structure of* $S_{j+1}, \ldots, S_{j+m}$*. Then the forward interface* FI_j *is a d-sepset between* Π *and* Φ*.*

Proof:

For each $x \in FI_j$, since x is the tail of one or more temporal arcs, the parent set $\pi(x)$ of x is contained in Π, and no parent of x is contained in Φ. By the definition of d-sepset, FI_j is a d-sepset between Π and Φ. □

Next, we show that FI_j is superior to the backward interface with respect to the mechanism **M1**.

Proposition 7 *Let* Π *and* Φ *be defined as Proposition 6. If there exist no converging temporal arcs from* Π *to* Φ*, then* $|FI_j| \leq |BI_{j+1}|$*.*

Proof:

Suppose there are no converging temporal arcs from Π to Φ. Then for every two temporal arcs (x, y) and (u, v) such that $x, u \in FI_j$ and $y, v \in BI_{j+1}$, $x \neq u$ implies $y \neq v$. Hence $|FI_j| \leq |BI_{j+1}|$. If there exists a temporal arc (x, y) and a node $z \in \pi(y)$ such that z is not the head of a temporal arc from Π to Φ, then $|FI_j| < |BI_{j+1}|$. □

Proposition 7 implies that if we use FI_i as the interface and complete FI_j instead of BI_{j+1}, then we can reduce STSS under **M1**. Although we have not been able to prove the same effect under **M2**, it appears that since completing FI_j needs less number of fill-ins, it will also introduce less number of untriangulated cycles. Consequently, it will be better-off under **M2** as well.

For example, we can represent the DBN in Figure 2 (a) as a MSBN whose subnet is shown in Figure 5 (a) (assuming $m = 1$). The d-sepset with the preceding subnet is the forward interface $\{a, d, h\}$ and the d-sepset with the following subnet is the forward interface $\{A, D, H\}$. Using the above idea to complete FI_j before triangulating[5] the moral graph, we obtain the template JT in (b). There is a single linkage $\{a, d, h\}$ (shown as a heavy line) between the JT and the preceding JT, which shall be referred to as a *backward linkage*, and a single linkage $\{A, D, H\}$ to the following JT, which shall be referred to as a *forward linkage*. The linkage hosts for the two linkages are C_1 and C_2,

[5] Triangulation is involved in this example and several others to follow. In all these examples, the triangulation used in creating the LJF is performed using the *maximum cardinality search* [6].

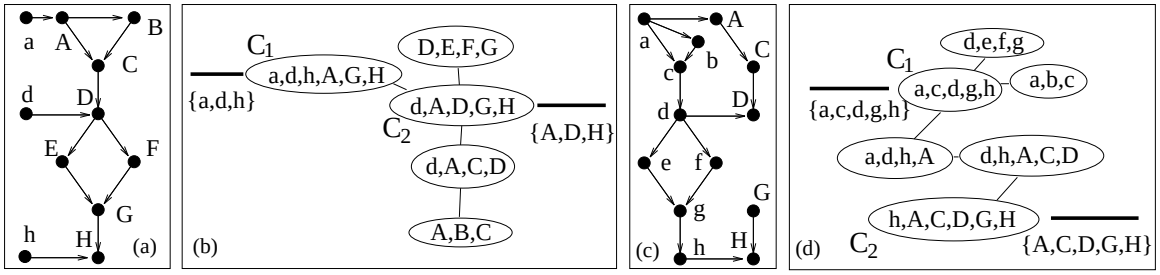

Figure 5: Representation of DBN using MSBN. (a) A subnet with forward interface as the d-sepset. (b) The template JT created from (a). (c) A subnet with backward interface as the d-sepset. (d) The template JT created from (c).

respectively. We shall refer to the method that uses FI_j as the d-sepset and completes FI_j before triangulation as *forward interface sectioning* or FI sectioning. We shall refer to the corresponding method using BI_{j+1} as *backward interface sectioning* or BI sectioning. Figure 5 (c) and (d) show a subnet and a resultant template JT obtained by BI sectioning with the backward interface $\{A, C, D, G, H\}$.

Comparing Figure 5 with Figure 2, clearly we have gained computational efficiency in two different ways. First, we no longer need to dynamically construct the JT in Figure 2 (c) from (a) and then reduce it to (d). We can simply use the JT in Figure 5 (b) or (d) repeatedly as described in Section 6. Second, using FI_j instead of BI_{j+1} as the d-sepset, we have cut down STSS from $2^6 * 1 + 2^5 * 2 + 2^4 * 2 + 2^3 * 1 = 168$ for Figure 5 (d) to $2^6 * 1 + 2^5 * 1 + 2^4 * 2 + 2^3 * 1 = 136$ for (b), which is 19% of cutback, assuming that all variables are binary as we will for all remaining examples.

The above cutback in the total state space is obtained by completing FI_j instead of BI_{j+1}, under the condition of no converging temporal arcs from Π to Φ. If a large number of converging temporal arcs exist, completing FI_j may still create a large clique. For such DBNs, we may improve FI sectioning by first converting each slice into one that has no converging temporal arcs. The conversion can be performed by introducing dummy variables as follows.

Procedure 8 *Let Π and Φ be defined as Proposition 6. Let $\mathcal{A}$ be the set of all converging temporal arcs from Π to Φ that share a common head x in Φ. Let π be the set of parents of x in Π.*

1. *Remove all temporal arcs in $\mathcal{A}$ from the DBN.*
2. *Add a dummy node x' in Π such that (x', x) is the only temporal arc from Π to Φ that has the head x.*
3. *Make π the parents of x' by adding arcs from π to x' in Π.*

Repeat the above steps for each pair of $\mathcal{A}$ and x.

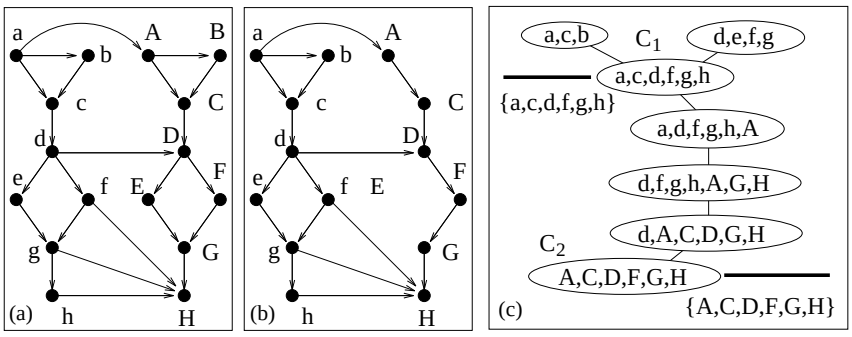

Figure 6: A DBN with converging temporal arcs. (a) Two adjacent slices (the temporal arcs into the first slice are not shown). (b) A subnet representation of the DBN in (a) using the backward interface as the d-sepset. (c) The template JT obtained from (b) by completing the backward interface.

Figure 6 (a) shows two slices of a DBN that has converging temporal arcs (f, H), (g, H) and (h, H). Figure 6 (b) and (c) show a subnet and a resultant template JT by BI sectioning. The STSS is $2^7 * 1 + 2^6 * 4 + 2^4 * 1 + 2^3 * 1 = 408$.

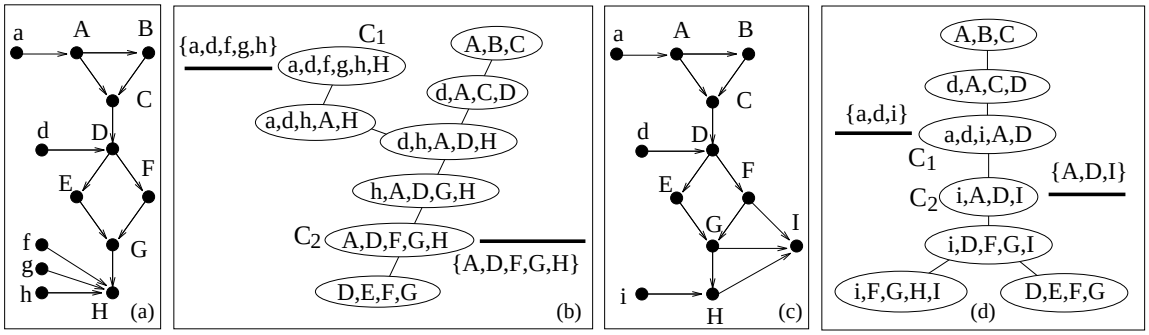

Figure 7: Representation of the DBN in Figure 6 (a). (a) A subnet representation using the forward interface as the d-sepset. (b) The template JT obtained from (a) by completing the forward interface. (c) A subnet representation with the dummy variables i and I introduced. (b) The template JT obtained from (c) by completing the forward interface.

Figure 7 (a) and (b) show a subnet and a resultant template JT by FI sectioning. The STSS is $2^6 * 1 + 2^5 * 4 + 2^4 * 2 + 2^3 * 1 = 232$. Figure 7 (c) and (d) show a subnet and a resultant template JT by FI sectioning with dummy variables introduced according to Procedure 8. Note that after the addition of dummy variables, there exist no converging

temporal arcs in the resultant DBN. The STSS is $2^5 * 3 + 2^4 * 3 + 2^3 * 1 = 152$, which is a 34% cutback from 232 obtained without using dummy variables and a 63% cutback from 408 obtained by BI sectioning.

8 Exploring Multiple Linkages

So far, we have tried to cut down STSS of the template JT by FI sectioning. FI sectioning works essentially by selecting the smallest possible d-sepset between adjacent subnets. With FI sectioning, in the best case, the size of the largest clique is no smaller than the number of non-converging temporal arcs (assuming converging arcs have been removed by Procedure 8). However, as this number increases, the complexity of the resultant template JT increases accordingly. Is it possible to further cut down the complexity?

Notice that FI sectioning passes the evidence in one JT to the next through a single linkage identical to the d-sepset. The CNE method is effectively doing the same thing. However, an important idea formalized in the theory of MSBNs is that the d-sepset may be decomposed into multiple linkages to cut down the computational expenses during evidence propagation between subnets (Section 5). We explore this idea below.

We first illustrate with an example and then generalize it. Consider the DBN in Figure 8 (a). If the CNE method or BI sectioning is used, a clique $C \supseteq \{A, C, D, G, H, K, L, O, P, S, T\}$ will be created. Since $|C| \geq 11$, the STSS of the resultant JT will be at least $2^{11} = 2048$. Figure 8 (b) and (c) shows a subnet and a resultant template JT obtained by FI sectioning. The resultant STSS is $2^8 * 3 + 2^7 * 2 + 2^4 * 9 + 2^3 * 1 = 1176$.

Figure 9 (a) shows a different subnet with the d-sepset $\{a, c, d, f, g, h, j, k, l, n, o, p, r, s, t\}$. Without forcing the completion of d-sepset before triangulation, we obtain the JT in (b). The backward linkages are shown in heavy dotted lines, and the forward linkages are shown in heavy solid lines. The corresponding backward and forward linkage trees are shown in (c) and (d). The STSS of the JT is $2^6 * 4 + 2^4 * 5 + 2^3 * 7 = 392$. By exploring the conditional independence within the d-sepset, we have obtained a 67% cutback from 1176 by FI sectioning, and at least 81% cutback from that by the CNE method or BI sectioning (using 2048).

We now analyze the conditions under which the above cutback can be obtained. Topologically, we notice that each slice of the DBN in Figure 8 (a) is 'narrow' horizontally (along the direction of temporal arcs) but is 'high' vertically. We shall loosely refer to the former direction as the *temporal direction* and to the latter as the *ortho-temporal direction*. We formalize the two scales as follows:

Definition 9 *Let S_i be a slice in a DBN and S_i' be its moral graph. Let x_{i-1}, x_i be a pair of nodes in S_i such that x_{i-1} is the tail of a temporal arc from FI_{i-1} to N_i and x_i is the tail of the corresponding temporal arc from FI_i to N_{i+1}.*

The `width` *of S_i* `measured at` *x_i is the length of the shortest path between x_{i-1} and x_i in S_i', denoted by $\tau(x_i)$. The* `width` *of S_i is $\tau = \max_{x_i}(\tau(x_i))$.*

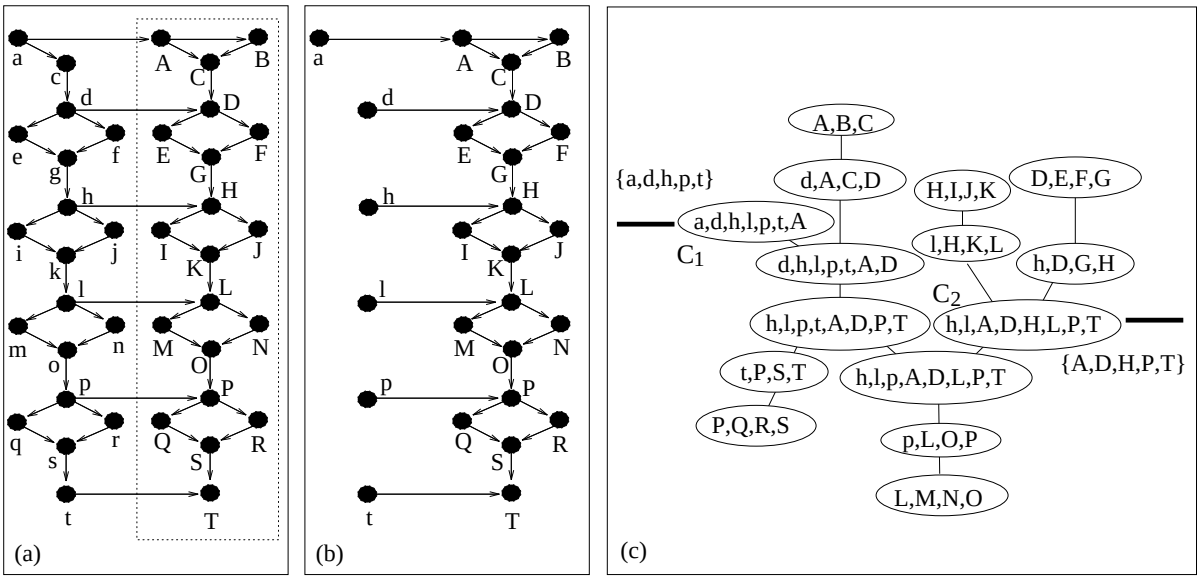

Figure 8: (a) A DBN. (b) A subnet representation of the DBN in (a). (c) A template JT obtained from (b) by completing the forward interface.

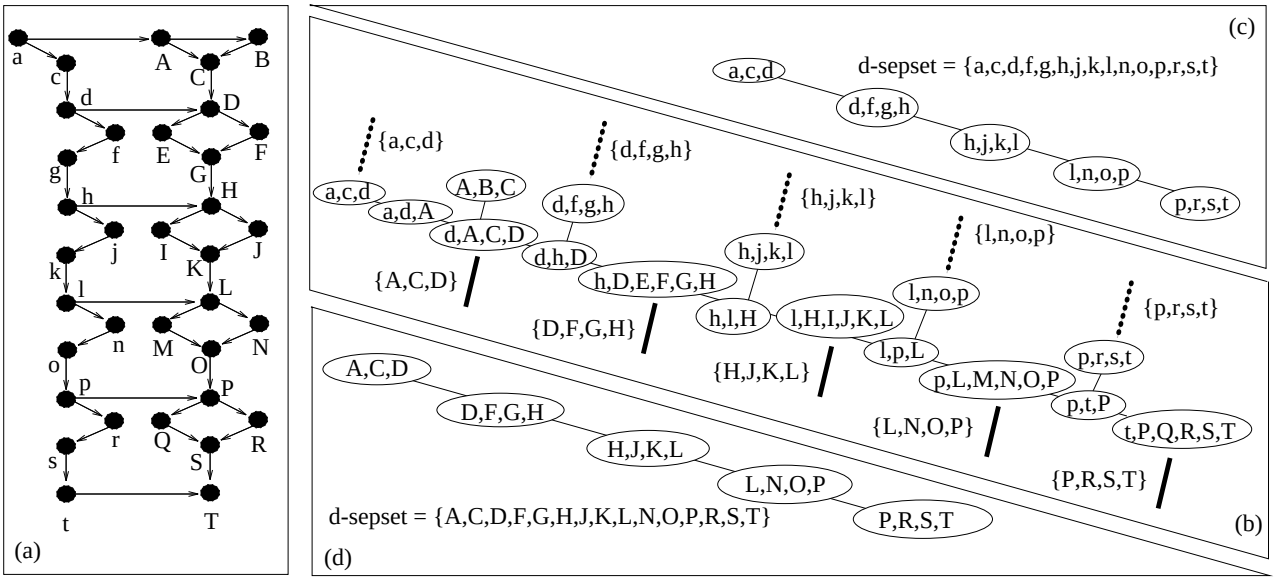

Figure 9: (a) A subnet representation of the DBN in Figure 8 (a). (b) The template JT obtained from (a). (c) The backward linkage tree of the JT in (b). (d) The forward linkage tree of the JT in (b).

The `height between` *a pair of nodes* y, z *of* S_i *is the length of the shortest path between them in* S_i'*, denoted by* $\rho(y, z)$*. The* `height` *of* S_i *is* $\rho = \max_{y,z}(\rho(y, z))$*.*

The `slimness` *of* S_i *is the ratio* ρ/τ*.*

For the DBN in Figure 8 (a), $\tau = 1$, $\rho = 15$, and $\rho/\tau = 15$. For the DBN in Figure 1, $\tau = 1$, $\rho = 4$, and $\rho/\tau = 4$. Therefore, the DBN in Figure 8 (a) is slimmer than the DBN in Figure 1.

When many temporal arcs exist between slices of a DBN, $|FI_j|$ is also large, and therefore FI sectioning generates large cliques as it completes FI_j. However, if the slice is slim, many small cutsets exist along the ortho-temporal directions, which correspond to conditional independencies that may be explored as in Figure 9. To this end, the ortho-temporal independencies must be preserved during sectioning. Sectioning in Figure 8 (b) does not preserve the independencies since the structure between nodes in the forward interface $\{a, d, h, l, p, t\}$ is deleted. In Figure 9 (a), the independencies among nodes in the forward interface are preserved by including additional d-sepnodes. Consequently, although the size of a subnet becomes larger, STSS of the template JT becomes much smaller.

9 Conclusions

We have proposed several techniques to improve the efficiency of exact inference in stationary DBNs.

First, when a DBN is stationary, the MSBN framework can be applied to create a template JT as a temporally invariant representation. This removes the need for the computation associated with dynamic expansion and reduction by the CNE method.

Second, we showed that the forward interface is usually a better d-sepset than the backward interface in that FI sectioning usually results in a smaller STSS than BI sectioning.

Third, we showed that when the DBN is slim, we may explore the conditional independencies along the ortho-temporal direction by preserving the independencies in the d-sepset and then decomposing the d-sepset into multiple linkages. This may reduce the STSS significantly compared with the CNE method or FI sectioning.

Our analysis provides the following guidelines in sectioning a DBN into a MSBN. If only a few temporal arcs exist between slices, FI sectioning is a reasonable alternative. The benefit over the CNE method is mainly the saving of expansion and reduction but not the cutback of the STSS. In this case, using more than one active slice does not change the single linkage and hence the computational expense is only proportional to the number of active slices.

On the other hand, if many temporal arcs exist between slices and slices are slim, exploration of ortho-temporal independencies not only saves the dynamic expansion and reduction, but can cut down the STSS significantly. In this case, it is not wise to use

more than one active slice. As more active slices are added, the subnet of the MSBN will become 'fat' (we can extend the definition of slimness of a slice to that of a subnet although we will not do so due to space limit). The linkage tree will quickly become a single huge clique and the efficiency that might be gained by exploring ortho-temporal independencies will diminish quickly.

Our approach has the potential to be extended to some non-stationary DBNs. If the DBN can be expressed by a small number of distinct slices, several template JTs may be created for each distinct slice, one for each distinct preceding slice. We may also relax the assumption of a constant number of active slices (Section 6) in the same way.

As the contribution presented represents our initial attempt and ongoing research to apply the MSBN framework to inference in DBNs, the results are preliminary. Several directions of future research can be identified: The FI sectioning has been proven to be superior than BI sectioning under the mechanism **M1**. Although it appears that it should be superior under **M2** as well, a rigorous justification is needed.

The definition of slimness of a slice is intuitive, but more in-depth analysis is required to better understand the relation between the slimness (or some better measurement) and the potential benefit from exploring ortho-temporal independencies.

The results of the paper are illustrated using constructed DBNs. Experimental studies using realistic DBNs are needed to verify the effectiveness of these results.

References

[1] S. Andreassen, R. Hovorka, J. Benn, K.G. Olesen, and E.R. Carson. A model-based approach to insulin adjustment. In *Proc. Third Conference on Artificial Intelligence in Medicine*, pages 239–248. Springer-Verlag, 1991.

[2] P. Dagum, A. Galper, and E. Horvitz. Dynamic network models for forecasting. In D. Dubois, M.P. Wellman, B. D'Ambrosio, and P. Smets, editors, *Proc. Eighth Conference on Uncertainty in Artificial Intelligence*, pages 41–48, Stanford, CA, 1992.

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

[4] T.L. Dean and M.P. Wellman. *Planning and Control*. Morgan Kaufmann, 1991.

[5] J. Forbes, T. Huang, K. Kanazawa, and S. Russell. The batmobile: towards a bayesian automated taxi. In *Proc. Fourteenth International Joint Conference on Artificial Intelligence*, pages 1878–1885, Montreal, Canada, 1995.

[6] M.C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Academic Press, 1980.

[7] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, (4):269–282, 1990.

[8] K. Kanazawa, D. Koller, and S. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In P. Besnard and S. Hanks, editors, *Proc. Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 346–351, Montreal, Canada, 1995.

[9] U. Kjaerulff. A computational scheme for reasoning in dynamic probabilistic networks. In D. Dubois, M.P. Wellman, B. D'Ambrosio, and P. Smets, editors, *Proc. Eighth Conference on Uncertainty in Artificial Intelligence*, pages 121–129, Stanford, CA, 1992.

[10] S.L. Lauritzen and D.J. Spiegelhalter. Local computation with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, (50):157–244, 1988.

[11] R.E. Neapolitan. *Probabilistic Reasoning in Expert Systems*. John Wiley and Sons, 1990.

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

[13] Y. Xiang. Optimization of inter-subnet belief updating in multiply sectioned bayesian networks. In *Proc. Eleventh Conference on Uncertainty in Artificial Intelligence*, pages 565–573, Montreal, Quebec, 1995.

[14] Y. Xiang, B. Pant, A. Eisen, M. P. Beddoes, and D. Poole. Multiply sectioned bayesian networks for neuromuscular diagnosis. *Artificial Intelligence in Medicine*, 5:293–314, 1993.

[15] 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.