Multiagent Bayesian Forecasting of Structural Time-Invariant Dynamic Systems with Graphical Models

Yang Xiang[a,*], James Smith[b], Jeff Kroes[c]

[a]*Department of Computing & Information Science, University of Guelph, Canada*
[b]*Department of Statistics, University of Warwick, UK*
[c]*Department of Computing & Information Science, University of Guelph, Canada*

Abstract

Time series are found widely in engineering and science. We study forecasting of stochastic, dynamic systems based on observations from multivariate time series. We model the domain as a dynamic multiply sectioned Bayesian network (DMSBN) and populate the domain by a set of proprietary, cooperative agents. We propose an algorithm suite that allows the agents to perform one-step forecasts with distributed probabilistic inference. We show that as long as the DMSBN is structural time-invariant (possibly parametric time-variant), the forecast is exact and its time complexity is exponentially more efficient than using dynamic Bayesian networks (DBNs). In comparison with independent DBN-based agents, multiagent DMSBNs produce more accurate forecasts. The effectiveness of the framework is demonstrated through experiments on a supply chain testbed.

Key words: Time series, forecasting, probabilistic inference, multiagent systems, graphical models, Bayesian networks, multiply sectioned Bayesian networks

1. Introduction

Many application domains in engineering, science and economics are complex, stochastic, and dynamic systems. For an intelligent agent to act effectively in such a domain, it is often necessary to predict the future development based on past observations. We model these dynamic systems as multi-dimensional stochastic processes, and we refer to *dynamic systems*, *dynamic domains*, and *multi-dimensional stochastic processes*, interchangeably. When observed, a multi-dimensional stochastic processes gives rise to a discrete time, multivariate time series [1, 2]. The primary inference that we address is one-step-ahead forecasting of these systems from the corresponding time series.

*Corresponding author
Email addresses: `yxiang@cis.uoguelph.ca` (Yang Xiang), `J.Q.Smith@warwick.ac.uk` (James Smith), `jkroes@uoguelph.ca` (Jeff Kroes)

Multivariate time series, especially economic time series, are commonly analyzed using the vector autoregressive (VAR) models [1]. These models relate the current value of a series to values in the past through autoregressive coefficients which do not vary with time.

Alternatively, time series are represented by state-space models, also referred to as multivariate dynamic linear models (DLMs) [3]. These are full probabilistic versions of the well known Kalman Filter, which is formally defined only in terms of the first and second moments of the process. Like the Kalman filter, they admit fast simple closed form recurrences, linking one-step-ahead forecast distributions to past observations. DLMs distinguish states of dynamic systems and observations that are dependent on the states. The evolution of states is encoded by the state equation (or system equation), and the dependency of observations on states is encoded by the observation equation. The class of VAR models are formally a subclass of multivariate DLMs, which generalize VAR models so that their natural autoregressive parameters are allowed to be dynamically changing.

Graphical models have become important tools in time series analysis in the last two decades, enhancing both VAR and state-space modeling. Dahlhaus and Elchler [4] consider two classes of graphical VAR models. In the first class, each observation variable at each specific time is represented as a node in the graph. In the second class, each univariate series is represented as a node. Both classes are mixed graphs with both directed and undirected links.

The state-space modeling is enhanced mostly by extending static Bayesian networks (BNs) [5] into DBNs [6]. In a DBN, each state or observation variable at each specific time is represented as a node in a directed acyclic graph (DAG). Each variable is associated with a conditional probability distribution that quantifies the strength of its dependency on its parent variables. DBNs therefore give a convenient qualitative framework that provides a compact representation of independence relationships that lie at the core of standard multivariate DLMs. This can be used to elegantly extend DLMs to non-Gaussian, non-linear domains. Exact inference in DBNs, including monitoring and forecasting, is often carried out by some form of elimination and message passing, e.g., [7, 8]. However, although the prior joint distribution over all variables in a DBN is factorized, the message to be passed during elimination is not factorable (Proposition 1 in [8]), and the message size is exponential on the number of persistent state variables. Approximate inference in DBNs has thus been preferred, e.g. [9, 10], in order to scale up.

An alternative class of graphical state-space models that allows exact forecasting without the disintegration of factorization is studied by Queen and Smith [11] and is termed multiregression dynamic models (MDMs). In MDMs, observation variables form nodes in a temporally extended DAG, while state variables are not represented in the graph. It was shown that if state variables are independent of each other a priori, they would remain so after observation. Furthermore, for linear MDMs, the first two moments of forecast distributions over observation variables can be calculated algebraically in closed form so that no approximation methods are necessary.

All above models assume a single-agent paradigm. To address the distribution of knowledge and data and to explore the benefit of distributed computation, analysis of time series under the multiagent paradigm has been seen in recent years. In [12], one-step-ahead forecasting is performed by a team of agents working for the same principal, and each agent is based on an artificial neural network with unique parameters. The agents compete to become a voting member and the forecast is determined through majority voting by voting agents. In [13], an agent plays the role of a manufacturer in a competitive market populated by self-interested agents and over a simulated year. The focus of the study is on one-step-ahead price forecasting by analysis of both the current time series (over the current year) as well as series from other years which generally involve different market conditions and different agents.

In both of the above multiagent systems, agents are competitive, although they work for the same principal in the former and for different principals in the latter. In [14], a dynamic system populated by cooperative agents is considered. The graphical models used by agents extend multiply sectioned Bayesian networks (MBSNs) [15] for static domains to dynamic domains. It does not consider time series with regularly spaced observations. Instead, a set of necessary observations needed to infer about a given subset of state variables is computed.

In this work, we consider one-step-ahead forecasting of a distributed, dynamic process with a cooperative multiagent system, fed by a distributed time series. Our approach extends the DBN-based graphical state-space modeling from single-agent to cooperative multiagent. Our forecasting algorithm suite belongs to the exact methods. Yet, in comparison with the equivalent DBN, our method improves the computational complexity significantly by reducing the total message size exponentially.

The remainder of the paper is organized as follows: Section 2 reviews the background on time series, DBNs and MSBNs. Section 3 defines the DMSBN representation of dynamic domains. Its multiagent adaptation is presented in Section 4 and is illustrated with an application. A number of properties of DMSBNs, including their structural and parametric time variability, are defined and analyzed in Section 5. A multiagent forecasting algorithm suite for time-invariant DMSBNs is presented in Section 6, and its exactness and complexity are analyzed in Section 7. How to transform parametric time-variant DMSBNs into time invariant DMSBNs is presented in Section 8. Our experimental results are reported in Section 9. We make concluding remarks in Section 10.

2. Background

In this section, we review the background and terminology on time series and three classes of graphical models: BNs for modeling a static domain under the single-agent paradigm, DBNs for modeling a dynamic domain under the single-agent paradigm, and MSBNs for modeling a static and distributed domain under the cooperative multiagent paradigm. In this work, variables in all graphical models considered are assumed discrete.

2.1. Bayesian Networks

A BN [5] typically models a static, stochastic domain.

Definition 1. *A* **BN** *is a triplet $\mathcal{G} = (V, G, P)$. V is a set of variables. G is a DAG whose nodes are labeled by elements of V. Each variable $v \in V$ is conditionally independent of its non-descendant variables in G given the set $\pi(v)$ of its parent variables. P is a set of conditional probability tables (CPTs) $P = \{P(v|\pi(v))|v \in V\}$.*

The joint probability distribution (JPD) over V is the product $P(V) = \prod_{v\in V} P(v|\pi(v))$.

2.2. Time Series

The behavior of a dynamic system is often recorded through a time series. In this work, we consider discrete time series over a finite time period T of $k+1$ time intervals (normally equally spaced). Below, we write a row vector as $(v_1, v_2, ...)$ and its transposition as $(v_1, v_2, ...)'$.

Definition 2. *A* **multivariate time series** *over time period $T = \{0, 1, ..., k\}$ is a multi-dimensional stochastic process of vectors $(x_{i1}, x_{i2}, ...)'$ observed at times $i \in T'$, where $T' \subseteq T$. Each $\{x_{ij}\}$ is a* **component** *series. A component series is* **complete** *if it contains a observation for each $i \in T$. Otherwise, it has* **missing values***.*

2.3. Dynamic Bayesian Networks

A DBN [6] models a dynamic system over $T = \{0, 1, ..., k\}$.

Definition 3. *A* **DBN** *of horizon k is a quadruplet*

$$\mathcal{G} = (\bigcup_{i=0}^{k} V_i, \bigcup_{i=0}^{k} G_i, \bigcup_{i=1}^{k} F_i, \bigcup_{i=0}^{k} P_i).$$

V_i is a set of variables for time interval i. G_i is a DAG whose nodes are labeled by elements of V_i. F_i is a set of arcs each directed from a node in G_{i-1} to a node in G_i. Each $v \in \bigcup_{i=0}^{k} V_i$ is conditionally independent of its non-descendants given its parents $\pi(v)$. P_i is a set of CPTs $P_i = \{P(v|\pi(v))|v \in V_i\}$.

$\mathcal{G}$ models a dynamic system whose condition at time interval i is represented by the set of variables V_i. The cardinality of V_i is assumed independent of time i, namely $|V_i| = |V_j|$ for $i \neq j$, and we denote $\eta = |V_0|$. The collection of the mth variable in V_0 through V_k forms a one-dimensional stochastic process. That is, for each $m = 1, ..., \eta$, the collection of variables $\{v_{0m}, ..., v_{km}\}$ where $v_{im} \in V_i$ forms a one-dimensional stochastic process. Hence, $\mathcal{G}$ models the multi-dimensional stochastic process

$$SP_T = (v_{i1}, ..., v_{i\eta})' \quad (v_{ij} \in V_i, i \in T).$$

Normally, V_i is partitioned into two sets: a set Y_i of state variables that are generally unobservable and a second set X_i of sensor variables that are observed but not necessarily at each time interval. The dependence and independence relations among variables in V_i are represented by the graph G_i. The state transition from time $i-1$ to i is represented by the set F_i of *temporal* arcs normally between state variables, i.e., from Y_{i-1} to Y_i. Since no arc directly connects Y_{i-j} and Y_i for $j > 1$, SP_T is assumed to be a first-order Markov process. The strength of dependency signified by G_i and the uncertainty of transition signified by F_i are quantified by CPTs in P_i. Observations over X_i form a time series, i.e.,

$$TS_T = (x_{i1}, x_{i2}, ...)' \quad (x_{ij} \in X_i, i \in T' \subseteq T)$$

is a multivariate time series.

Figure 1 shows the DAG structure of a DBN, where $V_1 = \{a_1, b_1, c_1, d_1, e_1, f_1\}$, the arcs in G_1 are $E_1 = \{(a_1, b_1), (b_1, d_1), (c_1, e_1), (d_1, e_1), (e_1, f_1)\}$, and $F_1 = \{(a_0, b_1), (f_0, f_1)\}$. Note that G_1 and G_2 are not isomorphic and we return to this issue in Section 5.

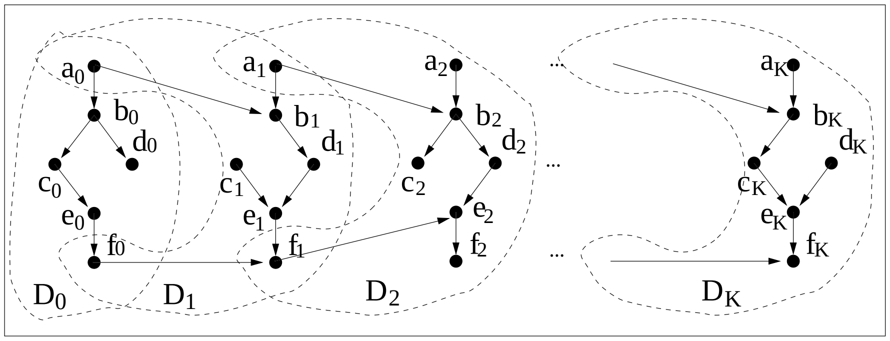

Figure 1: The structure of a DBN.

From Def. 3, the JPD over $V = \bigcup_{i=0}^{k} V_i$ is the product

$$P(V) = \prod_{i=0}^{k} \prod_{v \in V_i} P(v|\pi(v)).$$

Exact inference with a DBN can be performed in the same way as with a BN, but both the time and space complexity will be $O(k)$. The complexity can be reduced to being independent of k by recursively eliminating a historical portion of the DBN and passing a message with relevant information into a future portion. For example, one of the earliest exact methods [7] keeps only a segment of the DBN in memory, dynamically adds a new segment to the current segment, converts the expanded segment into a cluster tree, removes a part of the cluster tree corresponding to history, and uses the reduced cluster tree for inference.

In [8], the above dynamic expansion and reduction are replaced by a precompiled cluster tree template reused repeatedly during inference, further improving the efficiency. At each recursive step, before the cluster tree in the memory is discarded, it sends a message to the newly loaded template. The message is absorbed into the template which is then used to process new observations. This approach will be extended in Section 6 for multiagent forecasting. Below, we define components of a DBN used in the template and message computation.

Definition 4. *In a DBN $\mathcal{G}$ of horizon k, subset $FI_i = \{x | \exists\ (x, y) \in F_{i+1}\}$ is the* **forward interface** *of V_i $(0 \leq i < k)$. Denote $G_i = (V_i, E_i)$, where E_i is the set of arcs, and $D_i = (V_i \cup FI_{i-1}, E_i \cup F_i)$. The pair $S_i = (D_i, P_i)$ is the* **slice** *of the DBN for time i and D_i is the* **structure** *of S_i.*

In Figure 1, $FI_1 = \{a_1, f_1\}$, and $D_1 = \{a_0, f_0, a_1, b_1, c_1, d_1, e_1, f_1\}$. Each slice is enclosed in a dashed frame. Note that the slice of time i includes the forward interface from time $i - 1$. Note also that the first subscript is used to index temporal distribution of variables and dependency structures.

2.4. Multiply Sectioned Bayesian Networks

An MSBN [15] typically models a static, spatially distributed domain. By adequately decomposing the domain, an MSBN supports exact, distributed inference about the domain by a set of cooperative agents.

The domain dependence relations are represented distributively by a set of (overlapping) graphs. The terminology used to describe the relation among these graphs is defined below.

Definition 5. *Let $G^i = (V^i, E^i)$ $(i = 0, 1)$ be two graphs. G^0 and G^1 are* **graph-consistent** *if subgraphs of G^0 and G^1 spanned by $V^0 \cap V^1$ (keeping nodes in $V^0 \cap V^1$ and arcs among them only) are identical. Given two graph-consistent graphs $G^i = (V^i, E^i)$ $(i = 0, 1)$, the graph $G = (V^0 \cup V^1, E^0 \cup E^1)$ is the* **union** *of G^0 and G^1, denoted by $G = G^0 \cup G^1$.*

Given a graph $G = (V, E)$, a decomposition of V into V^0 and V^1 such that $V^0 \cup V^1 = V$ and $V^0 \cap V^1 \neq \emptyset$, and subgraphs G^i $(i = 0, 1)$ of G spanned by V^i, G is said to be **sectioned** *into G^0 and G^1.*

In Figure 2, the graph G to the right is the union of graphs G^0 and G^1 on the left. On the other hand, G is sectioned into G^0 and G^1.

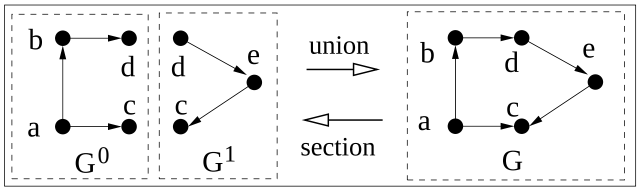

Figure 2: Illustration of graph union and section.

To support exact, distributed probabilistic inference, the stochastic domain as well as its dependency structure are decomposed according to the following conditions. Def. 6 specifies graph-theoretically how domain variables and their dependence structure are decomposed and distributed into a hypertree.

Definition 6. *Let $G = (V, E)$ be a connected graph sectioned into subgraphs $\{G^i = (V^i, E^i)\}$. Let the subgraphs be organized into an undirected tree Ψ where each node is uniquely labeled by a G^i and each link between G^k and G^m is labeled by the non-empty* **interface** *$V^k \cap V^m$ such that for each G^i and G^j in Ψ and each G^x on the path between G^i and G^j, $V^i \cap V^j \subset V^x$. Then Ψ is a* **hypertree** *over G. Each G^i is a* **hypernode** *and each interface is a* **hyperlink**. *A pair of hypernodes connected by a hyperlink is said to be* **adjacent**.

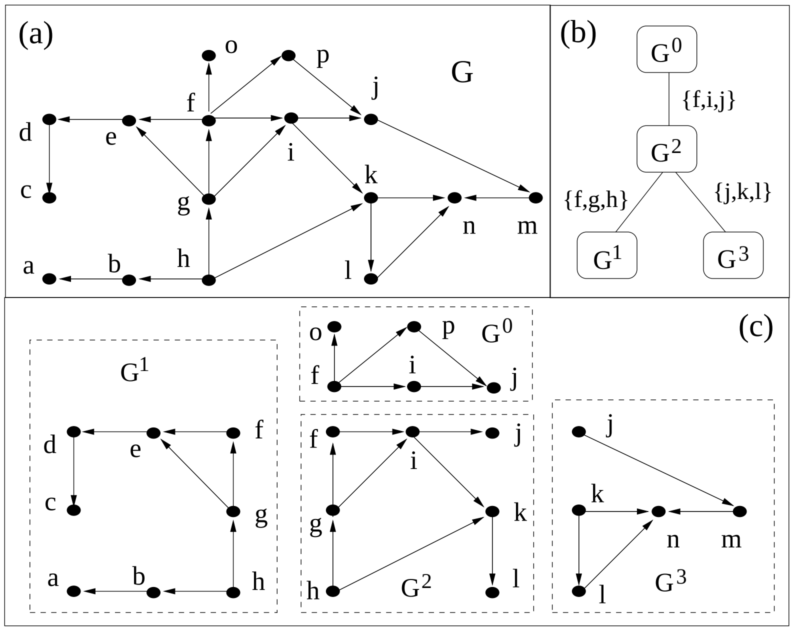

Figure 3: Graph G in (a) is sectioned into subgraphs in (c) with the hypertree in (b).

Figure 3 illustrates the sectioning of graph G (a) into the four subgraphs (c). The corresponding hypertree is shown in (b). The interface between hypernodes G_1 and G_2 is the set $\{f, g, h\}$. Note that the hypertree of Def. 6 satisfies the running intersection property.

Def. 6 only specifies the composition of interfaces. Def. 7 below further constrains the structure within an interface. This condition ensures that an interface induces conditional independence through d-separation [5].

Definition 7. *Let G be a directed graph sectioned into subgraphs $\{G^i\}$ such that a hypertree over G exists. A node x (whose parent set in G, possibly empty, is* *denoted $\pi(x)$) contained in more than one subgraph is a* **d-sepnode** *if there exists at least one subgraph that contains $\pi(x)$. An interface I is a* **d-sepset** *if every $x \in I$ is a d-sepnode.*

If a node x occurs in both G^a and G^b ($a \neq b$), its parent set $\pi^a(x)$ in G^a may differ from its parent set $\pi^b(x)$ in G^b, as well as from its parent set $\pi(x)$ in G. In Figure 3, $\pi^0(i) = \{f\}$ while $\pi^2(i) = \{f, g\} = \pi(i)$. The interface $\{f, i, j\}$ between G_0 and G_2 is a d-sepset because $\pi(f)$ and $\pi(i)$ are contained in G_2, and $\pi(j)$ is contained in G_0.

Def. 8 combines the above definitions to specify the dependence structure of an MSBN.

Definition 8. *A* **hypertree MSDAG** $G = \bigcup_i G^i$*, where each G^i is a DAG, is a connected DAG such that (1) there exists a hypertree Ψ over G, and (2) each hyperlink in Ψ is a d-sepset.*

Def. 9 defines an MSBN. Note that the superscript is used to index spatial distribution of variables and dependency structures.

Definition 9. *An* **MSBN** *M is a triplet $M = (V, G, P)$. $V = \bigcup_i V^i$ is the* **domain** *where each V^i is a set of variables, called a* **subdomain**. *$G = \bigcup_i G^i$ (a hypertree MSDAG) is the* **structure** *where nodes of each DAG G^i are labeled by elements of V^i. Each $x \in V$ is conditionally independent of its non-descendants given its parents $\pi(x)$ in G.*

$P = \bigcup_i P^i$ is a collection of CPTs, where $P^i = \{P(x|\pi(x)) | x \in V^i\}$, subject to the following condition: For each x, exactly one of its occurrences (in a G^i containing $\{x\} \cup \pi(x)$) is associated with $P(x|\pi(x))$, and each occurrence in other DAGs is associated with a constant (uniform) CPT.

Each triplet $S^i = (V^i, G^i, P^i)$ is called a **subnet** *of M. Two subnets S^i and S^j are* **adjacent** *if G^i and G^j are adjacent on the hypertree.*

From Def. 9, the JPD over V is the product

$$P(V) = \prod_i \prod_{v \in V^i} P(v|\pi(v)).$$

An MSBN models the domain V through subnets over its subdomains. When these subdomains are naturally distributed, they can be populated by multiple agents so that each subnet is embodied by a distinct agent. We refer to the MSBN associated with these agents as a *multiagent MSBN*. The multiagent MSBN enables agents to reason about the domain by distributed inference. The hypertree defines the agent organization and specifies the direct communication links among agents. For instance, let Figure 3 be the structure of a (trivial) MSBN populated by agents A^0 through A^3. According to the hypertree, A^0 directly communicates with A^2 only. The interfaces in the hypertree define the content of messages between agents, and are referred to as *agent interfaces*.

For exact, distributed inference by a multiagent system, each subnet is compiled into a local junction tree (JT), which is a cluster tree that satisfies the running intersection property. Each cluster is associated with a potential defined from CPTs in the corresponding subnet. These local JTs are linked according to corresponding agent interfaces. The multiagent MSBN is thus compiled into a linked junction forest (LJF) and each local JT is embodied by an agent. To reason about the domain, an agent can perform a local operation **UnifyBelief** to update its belief over its subdomain relative to its own observations after the last communication. The operation involves two rounds of message passing in the local JT. Furthermore, communication can be initiated by any agent and involve all agents through the operation **CommunicateBelief**, which updates the belief of each agent over its subdomain relative to observations made by all agents. The operation involves two rounds of message passing along the hypertree and each message between two agent is a potential over their agent interface. Details on these operations can be found in [15].

3. Dynamic Multiply Sectioned Bayesian Networks

In this section, we consider the modeling of a domain that is both dynamic and spatially distributed. At any given time, the domain is decomposed into overlapping subdomains. The temporal evolution of each subdomain is represented by a DBN (to be formalized in Section 5) which can be embodied by a distinct agent. The slices at time interval i from all DBNs form an MSBN (to be formalized in Section 5) and represent the condition of the distributed domain. We first define such a model formally as a DMSBN. We present its multiagent adaptation in Section 4 illustrated with an application. Several important properties of DMSBNs are defined and analyzed in Section 5. In the following, the first subscript is used to index the temporal evolution and the first superscript is used to index the spatial distribution.

Definition 10. *A* **DMSBN** *DM of horizon k is a quadruplet*

$$DM = (\bigcup_{i=0}^{k} V_i, \bigcup_{i=0}^{k} G_i, \bigcup_{i=1}^{k} F_i, \bigcup_{i=0}^{k} P_i).$$

$V_i = \bigcup_j V_i^j$ is the **domain** *for time interval i, where V_i^j is a* **subdomain** *for time i. $G_i = \bigcup_j G_i^j$ (a hypertree MSDAG) is the* **structure** *for time i, where nodes of each DAG $G_i^j = (V_i^j, E_i^j)$ are labeled by elements of V_i^j. $F_i = \bigcup_j F_i^j$ is a collection of* **temporal arcs***, where F_i^j is a set of arcs each directed from a node in G_{i-1}^j to a node in G_i^j. Each $v \in \bigcup_{i=0}^k V_i$ is conditionally independent of its non-descendants given its parents $\pi(v)$ in $\bigcup_{i=0}^k G_i$.*

$P_i = \bigcup_i P_i^j$ is a collection of CPTs, where $P_i^j = \{P(x|\pi(x))|x \in V_i^j\}$, subject to the following condition: For each $x \in V_i$, exactly one of its occurrences (in a G_i^j containing $\{x\} \cup \pi(x)$) is associated with $P(x|\pi(x))$, and each occurrence in other DAGs for time i is associated with a constant CPT.

The **forward interface** *of subdomain V_i^j $(0 \leq i < k)$ is $FI_i^j = \{x | \exists\, (x, y) \in F_{i+1}^j\}$ with $FI_{-1}^j = \emptyset$.*

The jth **subnet** *of DM for time i is a triplet $S_i^j = (\hat{V}_i^j, \hat{G}_i^j, \hat{P}_i^j)$. Its (enlarged) subdomain is $\hat{V}_i^j = V_i^j \cup FI_{i-1}^j$. Its (enlarged) subnet structure is $\hat{G}_i^j = (\hat{V}_i^j, \hat{E}_i^j)$, where $\hat{E}_i^j = E_i^j \cup F_i^j$. The set of CPTs (one per node) in the subnet is $\hat{P}_i^j = \{P(x|\pi(x))|x \in \hat{V}_i^j\}$ except that each $x \in FI_{i-1}^j$ is assigned a constant CPT.*

A **slice** *of DM for time i is*

$$M_i = \bigcup_j S_i^j = (\bigcup_j \hat{V}_i^j, \bigcup_j \hat{G}_i^j, \bigcup_j \hat{P}_i^j).$$

The condition of the dynamic system at time i is represented by $V_i = \bigcup_j V_i^j$ and the cardinality of V_i^j is assumed independent of i. Each collection of variables $\{v_{0m}^j, v_{1m}^j, ..., v_{km}^j\}$, where $v_{im}^j \in V_i^j$ and $1 \leq m \leq |V_0^j|$, forms a one-dimensional stochastic process. Denote $\eta = |V_0|$. DM models a multi-dimensional first-order Markov stochastic process

$$SP_T = (v_{i1}, ..., v_{i\eta})' \quad (v_{im} \in V_i, i \in T).$$

The JPD over $V = \bigcup_{i=0}^k V_i$ is

$$P(V) = \prod_{v \in V} P^*(v|\pi(v)),$$

where $P^*(v|\pi(v)) = P(v|\pi(v))$ if v occurs in a unique V_i^j. Otherwise, $P^*(v|\pi(v))$ equals $P(v|\pi(v))$ associated with the occurrence of v that is assigned a non-constant CPT.

Def. 10 defines a DMSBN based on the forward interface. This is not necessary as our results apply to other temporal interfaces as well, such as backward interface [8].

4. Multiagent DMSBNs and an Application

Although not required by Def. 10, DMSBNs are particularly useful for modeling a dynamic domain that is spatially or otherwise distributed and can benefit from multiagent processing. Under the multiagent paradigm, the domain is populated by a set of agents. Each agent A^j is in charge of the subdomain V_i^j at time i and embodies the subnet S_i^j for $i = 0, 1, ..., k$. That is, A^j is associated with the multi-dimensional stochastic process

$$(v_{i1}, v_{i2}, ...)' \quad (v_{im} \in V_i^j, i \in T).$$

To take advantage of the interdependence between processes at different agents, at any time i, subdomains are organized into a hypertree, and agents can communicate through interfaces specified by the hypertree. We refer to an interface on the hypertree as an *agent interface*.

We assume that agents generally work for different principals. Therefore, associated with the spatial distribution of variables, there is also the distribution of ownership and interest. However, agents working for different principals do not have to be competitive. Two assumptions are often made for DMSBN-based multiagent systems. The first is the *proprietary* assumption: The knowledge of A^j over V_i^j and S_i^j is proprietary. Hence, variables in V_i^j that are not contained in any agent interface of A^j are private variables of A^j. The dependency structure among them as well as numerical parameters that quantify the structure are also private to A^j. As a result, a centralized representation and processing of a DMSBN is not feasible.

The second assumption is *common interest*: Agents share a common interest that motivates them to cooperate truthfully within the limit of their privacy. Hence, agents can form agreeable interfaces, and variables contained in agent interfaces are public. Furthermore, any message exchanged regarding public variables is consistent with the true belief of the sending agent. No messages regarding private variables will be communicated.

We refer to a DMSBN as a *multiagent DMSBN* if it is populated by a set of agents to whom the proprietary and common interest assumptions hold. We consider below an application of multiagent DMSBNs.

Supply chain forecasting Manufacturers in a supply chain are related by recursive supplier-consumer relations. For each supplier to meet the needs of production operations for workers (to be hired or laid-off), equipment (to be purchased or reconfigured), materials (to be ordered and shipped) and so on, arrangements must often be made in advance. Forecasts allow such needs to be anticipated so that necessary arrangements are made in time.

More specifically, consider the issue from the equipment perspective. Manufacturing of a particular part, device, or component requires setup and reconfiguration of equipment. Per-part cost is reduced if setup is performed once for a large batch of the same part. Constant switching between manufacturing of different parts increases per-part cost and should be avoided. On the other hand, if production exceeds demand significantly, maintaining a large inventory over an extended period is also costly. Hence, accurate prediction of short-term demand allows the optimal planning of manufacturing operations.

Supply chain forecast can be performed by a multiagent system where each manufacturer is served by a computational agent. These agents can model the domain as a DMSBN. Figure 4 illustrates a DMSBN for a three-agent system with $T = \{0, 1\}$. The spatial distribution is shown along the horizontal direction. For example, G_0^0, G_0^1 and G_0^2 for time 0 are shown in dashed boxes on the top row, and the corresponding agents A^0, A^1 and A^2 are indicated. These graphical structures encode the following dependence relations. For each supplier, availability of skilled workers, adequate equipment, and material (or component) ordered constrain the level of production, which in turn determines the amount of supply produced and influences the unit cost. Availability of skilled workers influences the workers' wage, which in turn affects the unit cost. The unit cost is also affected by the sale price of the material from the next supplier down the chain. The amount of supply and the order incoming from

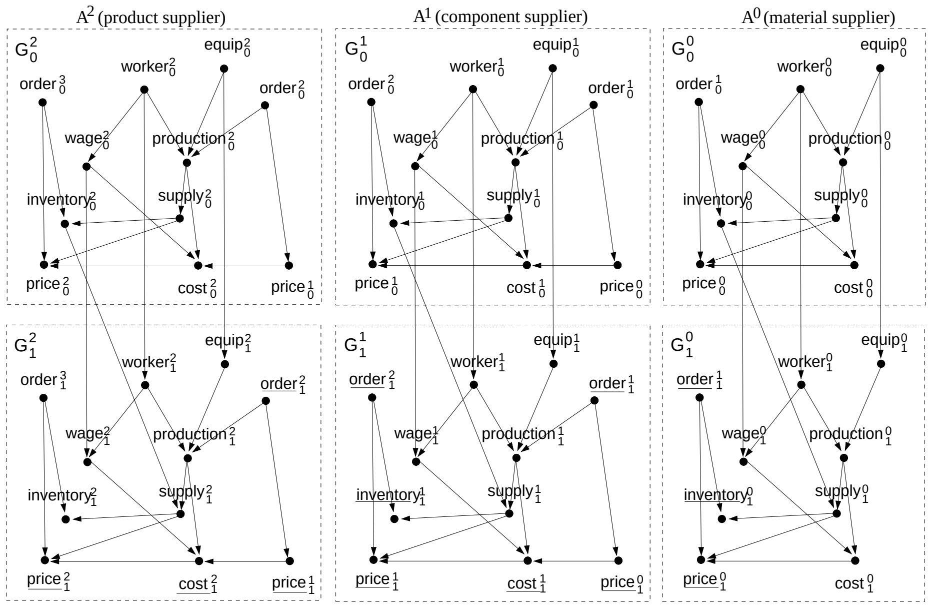

Figure 4: The structure of a DMSBN for supply chain forecasting.

the next supplier up the chain determine the inventory left and affect the unit sale price. Note that $production_0^1$ is a private variable for agent A^1, while $price_0^1$ is a public variable between A^1 and A^2.

The temporal evolution is shown vertically. For example, G_1^2 (bottom left) is placed below G_0^2. The set F_1^2 of temporal arcs consists of those going from the upper left box into the bottom left box. They encode the following temporal dependency. The current availability of workers, the current wage level, and the current availability of equipment are dependent on their status in the previous time. The current supply is also affected by the previous level of inventory.

In supply chains, the demand (from a consumer) of a component (satisfied by a supplier) generates a demand of parts (produced by other suppliers) that the component is composed of. This interdependency among suppliers makes isolated forecasting by individual manufacturers less accurate. A cooperative forecasting is advantageous as each agent benefits from knowledge and observations of other agents over their subdomains. More accurate forecasting allows better planning and more cost-effective operations for all suppliers. Hence, DMSBNs have an advantage over isolated DBNs as will be shown in Section 9.

5. Properties of DMSBNs

In general, as time evolves, the jth subnet of a DMSBN for any fixed j may change both its graphical structure and its parameters. We describe this property of DMSBNs with the time variability defined below.

Definition 11. *A DMSBN is* **structural time-invariant** *if for all $i \neq j$, G_i and G_j are isomorphic and F_i and F_j are isomorphic. Otherwise, it is* **structural time-variant***.*

In a structural time-invariant DMSBN, parameter sets P_i and P_j for some $i \neq j$ ($i > 0, j > 0$) are **equivalent** *if for every variable x_i in G_i and its isomorphic counterpart x_j in G_j, $P(x_i|\pi(x_i)) \in P_i$ is identical to $P(x_j|\pi(x_j)) \in P_j$.*

A structural time-invariant DMSBN is also **parametric time-invariant** *if for all $i \neq j$ ($i > 0, j > 0$), P_i and P_j are equivalent. Otherwise, the DMSBN is* **parametric time-variant***.*

A DMSBN is **time-invariant** *if it is both structural time-invariant and parametric time-invariant* [1]*.*

If a dynamic domain can be modeled by a structural time-invariant DMSBN, it must be the case that at any fixed time, the set of dependence relations (temporal or atemporal) do not change over time. Hence, graph substructures of the DMSBN are isomorphic. If the domain can be modeled by a time-invariant DMSBN, then in addition, relative to the dependence structure, the strengths of dependence relations do not change over time (CPTs are equivalent). Time-invariant DMSBNs can be more effectively processed than their time-variant counterpart because agents can reuse template subnets, as we explain in Sections 6 and 7. Although parametric time-variant DMSBNs appear to be out of reach by such template-based processing, we show in Section 8 that as long as they are structural time-invariant, they can be transformed into time-invariant DMSBNs and hence be amendable to such processing. We will refer to the class of multi-dimensional stochastic processes that can be modeled by structural time-invariant DMSBNs as *structural time-invariant processes*.

The time variability defined above should not be confused with stationarity and homogeneity, two commonly referred-to properties of stochastic processes. A stochastic process is *stationary* if the joint distribution of any collection of m variables of the process does not change when shifted in time [1, 3]. A process that can be represented by a time-invariant DMSBN may not be stationary. A Markov chain $\{v_0, v_1, ...\}$, for instance, generally has $P(v_1) = \sum_{v_0} P(v_1|v_0)P(v_0) \neq P(v_0)$ even if $P(v_{i+1}|v_i)$ does not change with i. Hence, time-invariability does not imply stationarity.

A Markov process is *homogenous* if the conditional probability distribution of its state transition does not change when shifted in time [16]. A DMSBN can be viewed as consisting of a state model (state variables in V_i and relevant part of G_i and P_i), a transition model (F_i and relevant part of P_i), and a sensor model (sensor variables in V_i and relevant part of G_i and P_i). Homogeneity focuses on invariability for the transition model, while time-invariability requires invariability for the state model and the sensor model as well. Hence, time-invariability is not equivalent to homogeneity.

[1]Note that a parametric time-invariant DMSBN is also a time-invariant DMSBN.

Below, we establish the fundamental relations between DBNs, MSBNs, and DMSBNs. This not only gains insight into the syntax and semantics of DMSBNs, but also suggests the application of inference methods for DBNs and MSBNs to Bayesian forecasting with DMSBNs. Proposition 1 establishes the relation between DBNs and DMSBNs. Its proof is straightforward by comparing Def. 3 and Def. 10.

Proposition 1. *Let DM be a DMSBN of horizon k. Then, for each j,*

$$DM^j = (\bigcup_{i=0}^{k} V_i^j, \bigcup_{i=0}^{k} G_i^j, \bigcup_{i=1}^{k} F_i^j, \bigcup_{i=0}^{k} P_i^j)$$

is a DBN.

Note that from Def. 10, for each variable x with multiple occurrences at time i, only one occurrence is associated with the non-constant $P(x|\pi(x))$ and each other occurrence is associated with a constant CPT. Hence, the product of CPTs at nodes in the DBN DM^j is not necessarily identical to the marginal of JPD from DM marginalized down to $\bigcup_{i=0}^{k} V_i^j$. This issue must be addressed when the method for inference in DBNs is extended to DMSBNs.

Proposition 2 establishes the relation between MSBNs and DMSBNs.

Proposition 2. *Let DM be a DMSBN of horizon k and M_i be a slice of DM for time i. Then M_i is an MSBN.*

Proof: The proof is straightforward by comparing Def. 9 and Def. 10 and noting the following: Although in each subnet S_i^j of M_i, G_i^j is enlarged into $\hat{G}_i^j$ with FI_{i-1}^j and F_i^j, the temporal arcs F_i^j do not introduce direct connection between G_i^j and G_i^m for all $m \neq j$. Hence, whenever $G_i = \bigcup_j G_i^j$ is a hypertree MSDAG, $\hat{G}_i = \bigcup_j \hat{G}_i^j$ is also a hypertree MSDAG. □

Note that for each $x \in FI_{i-1}^j$ in the subnet S_i^j, it has no parent in S_i^j and is assigned a constant CPT in Def. 10. Hence, $P(FI_{i-1}^j)$ as defined by S_i^j is a constant distribution as well. More precisely, the following marginalization

$$\sum_{\hat{V}_i \setminus FI_{i-1}^j} \prod_j \prod_{p(v|\pi(v)) \in \hat{P}_i^j} p(v|\pi(v)) \qquad (where\ \ \hat{V}_i = \bigcup \hat{V}_i^j)$$

is a constant distribution. This issue must be addressed when the method for inference in MSBNs is extended to DMSBNs. We summarize this fact in the following Lemma, which is needed in our later analysis.

Lemma 1. *Let DM be a DMSBN of horizon k and M_i be a slice of DM for time $i > 0$. Then, in each subnet, the distribution over the forward interface FI_{i-1}^j*

$$P(FI_{i-1}^j) = \sum_{\hat{V}_i \setminus FI_{i-1}^j} \prod_j \prod_{p(v|\pi(v)) \in \hat{P}_i^j} p(v|\pi(v)) \qquad (where\ \ \hat{V}_i = \bigcup \hat{V}_i^j)$$

is a constant distribution.

An important property of MSBNs is that they support distributed, exact inference by multiagent systems. This property results from the fact that the JPD $P(V)$ of the MSBN can be factorized according to subdomains. As shown in Proposition 1 from [8], although the prior joint distribution over all variables in a DBN is factorized, the message to be passed during inference by elimination is not factorable, and its size is exponential on the number of persistent state variables. Without a subdomain based factorization, exact inference in DMSBNs cannot be distributed to multiagent.

To resolve this issue, we require that the time series associated with the DMSBN satisfy the following *sufficiency* condition. It asserts that, before a forecast is made, all variables in the agent interface are observed.

Definition 12. *Let DM be a multiagent DMSBN over $T = \{0, ..., k\}$,*

$$DM = (\bigcup_{i=0}^{k} V_i, \bigcup_{i=0}^{k} G_i, \bigcup_{i=1}^{k} F_i, \bigcup_{i=0}^{k} P_i),$$

and Π_i be the set of all public variables in V_i. Let TS be a time series over $T^- = \{0, ..., k-1\}$,

$$TS = (x_{i1}, x_{i2}, ...)' \quad (x_{ij} \in X_i \subseteq V_i, i \in T' \subseteq T^-).$$

Then TS is a **time series associated** *with DM. TS is* **sufficient** *iff $\Pi_i \subseteq X_i$ and for each $x_{ij} \in \Pi_i$, the component series $\{x_{ij}\}$ is complete in T^-.*

In Def. 12, the DMSBN is over $k+1$ intervals while the time series is over k intervals. This sets the stage for multiagent forecasting for time k based on observations over the previous intervals. The sufficiency condition says that variables in agent interfaces must be observed without missing values. The relation $\Pi_i \subseteq X_i \subseteq V_i$ says that additional observations on private variables may be made and with missing values. We assume that the time series associated with a multiagent DMSBN satisfies the sufficiency condition. We refer to this as the *sufficient time series* assumption, or *agent interface observability* assumption. We show in the next two sections that this assumption enables exact forecasting by multiagent, distributed computation.

6. Multiagent Bayesian Forecasting in Time-Invariant DMSBNs

In this section, we consider multiagent one-step-ahead forecasting with a time-invariant multiagent DMSBN DM over T and a sufficient associated time series TS. To simplify the description, we assume that at each time $i \in T$, the observations in TS up to $i-1$ are available to agents before the forecast for time i is made. We also assume that no forecast is made for $i = 0$. The forecasting proceeds as follows:

At time $i = 0$, agents communicate through the MSBN M_0 to acquire prior distributions for their respective subdomains. That is, each agent A^j acquires the prior $P(\hat{V}_0^j)$ for $i = 0$. Then, each agent A^j makes a set of (local) observations obs_0^j in TS and updates belief about its subdomain $\hat{V}_0^j$ to get the posterior distribution $P(\hat{V}_0^j|obs_0^j)$ for $i = 0$. Due to the d-sepset condition of agent interface and the agent interface observability, this step can be performed at each agent's local JT without communication. By marginalizing $P(\hat{V}_0^j|obs_0^j)$, the posterior $P(FI_0^j|obs_0^j)$ over the forward interface is obtained. This is the message to be propagated into the subnet for $i = 1$.

After that, the MSBN M_1 is loaded into agents. At each agent A^j, the message $P(FI_0^j|obs_0^j)$ is absorbed into the subnet for $\hat{V}_1^j$. Through the MSBN M_1, agents communicate and make forecast for $i = 1$. That is, each agent A^j obtains the prior $P(\hat{V}_1^j|obs_0)$ for $i = 1$, where obs_0 includes all (global) observations in TS for $i = 0$.

From then on, at each time $i \geq 1$, each agent A^j acquires observations obs_i^j from TS, updates its belief into the posterior $P(\hat{V}_i^j|obs_0^j, ..., obs_i^j)$ through inference at its local JT, and obtains the marginal $P(FI_i^j|obs_0^j, ..., obs_i^j)$ over the forward interface. After that, the MSBN M_{i+1} is loaded into agents; the message $P(FI_i^j|obs_0^j, ..., obs_i^j)$ is absorbed at each agent A^j; agents communicate through M_{i+1} to obtain the prior $P(V_{i+1}^j|obs_0, ..., obs_i)$ for $i+1$ at each A^j; and the forecast for $i + 1$ is made accordingly.

Similar to inference in multiagent MSBNs, the above computation is best performed through a compiled representation of the DMSBN. The subnets for each time i are compiled into an LJF. As the DMSBN is assumed time-invariant, the LJF can be compiled once and reused for each time interval. The compilation is similar to that for MSBNs, except that for each subnet of time i, FI_{i-1}^j is contained in a cluster in the local JT and so is FI_i^j (see Proposition 1 in [8]). We denote the local JT of agent A^j compiled from its subnet S_i^j by T_i^j.

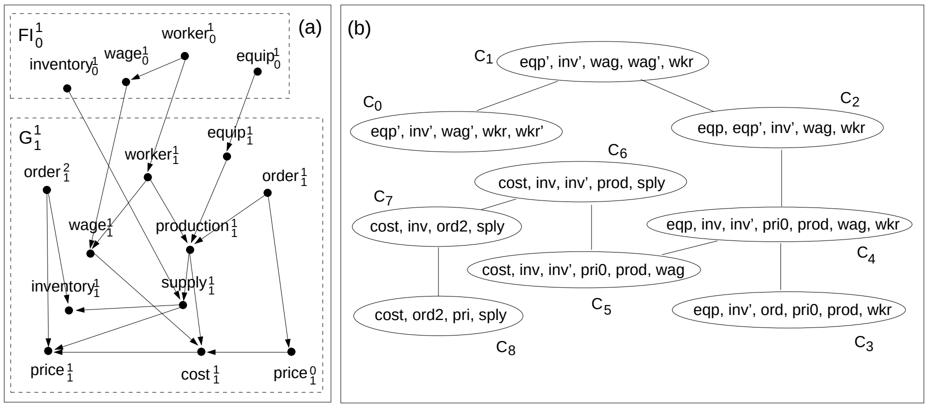

Figure 5: The structures of subnet S_1^1 (a) and local JT T_i^1 (b) for agent A^1 in Figure 4.

Figure 5 (a) shows the structure of subnet S_1^1 for the DMSBN illustrated in Figure 4. The structure of the corresponding local JT T_i^1 is shown in (b). Variable labels are simplified in (b). For variables in V_i^1 (lower box in (a)),

$wage_i^1$ is labeled as wag, $price_i^0$ as $pri0$, and $order_i^2$ as $ord2$. For variables in FI_{i-1}^1 (upper box in (a)), $wage_{i-1}^1$ is labeled as wag'.

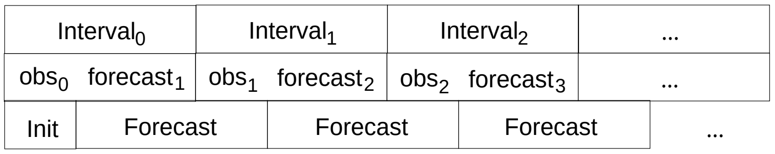

Figure 6: Timing of multiagent one-step-ahead forecasting.

Figure 6 illustrates the timing of agent activities. The first row shows the physical time intervals and their bounds. In the second row, the label obs_0 indicates the time when observations are made from time series TS for interval 0, and the label $forecast_1$ indicates the time when the forecast for time 1 is available. The overall computation is grouped into two algorithms **InitialObservation** and **Forecast** specified below. The third row illustrates the timing when each of the algorithms is executed in relation to the timing mentioned above, where *Init* stands for **InitialObservation**.

Algorithm 1 (InitialObservation). *At start of interval 0, each agent A^j does the following:*

```
1  load local JT $T_0^j$ into memory;
2  enter local observations from TS for time $0$;
3  perform UnifyBelief in $T_0^j$;
```

Algorithm 2 (Forecast). *At end of interval $i \geq 0$, each agent A^j does the following:*

```
1  retrieve potential $B(FI_i^j)$ from its local JT $T_i^j$;
2  replace $T_i^j$ by $T_{i+1}^j$ in memory;
3  find a cluster $Q$ in $T_{i+1}^j$ such that $Q \supseteq FI_i^j$;
4  absorb $B(FI_i^j)$ into potential $B(Q)$ with $B'(Q) = B(Q) * B(FI_i^j)$;
5  respond to call on CommunicateBelief;
6  make forecast for time $i + 1$;
```

During interval $i + 1$, A^j does the following:

```
7  enter local observations from TS for time $i + 1$;
8  perform UnifyBelief in $T_{i+1}^j$;
```

Note that for each $x \in FI_{i-1}^j$ in the subnet S_i^j, it has no parent in S_i^j and is assigned a constant distribution. Hence, $B(FI_{i-1}^j)$ in T_i^j is a constant distribution immediately after the local JT is loaded into memory.

CommunicateBelief is called upon an arbitrary agent during each interval. In the next section, we show that forecasts produced by a multiagent DMSBN based on the above algorithm suite are exact.

7. Exactness and Complexity of Forecasting

Theorem 1 below establishes that Bayesian forecasting in a multiagent DMSBN using **InitialObservation** and **Forecast** is exact.

Theorem 1. *After execution of* **InitialObservation** *at each agent, followed by* **Forecast** *from time interval* 0 *to* $i-1$*, followed by the first 6 lines of* **Forecast** *at the end of interval* i*, the one-step-ahead forecasts for time* $i+1$ *are exact.*

Before proving the theorem, we briefly introduce the necessary formal notions which are detailed in [15]. Recall from Def. 10, each subnet S_i^j in the MSBN M_i has the enlarged subdomain $\hat{V}_i^j$ and the structure $\hat{G}_i^j$. As is shown in the proof of Proposition 2, if $\cup_j G_i^j$ is a hypertree MSDAG, then $\cup_j \hat{G}_i^j$ is also a hypertree MSDAG. We denote the mth interface in this hypertree by $\hat{I}_i^m$.

Each subnet S_i^j is compiled into a local JT T_i^j with its potential $B_{T_i^j}(\hat{V}_i^j)$ defined through potentials over its clusters[2]. The argument in $B_{T_i^j}(\hat{V}_i^j)$ indicates the set of variables over which the potential is defined and the subscript, when used, emphasizes the object with which the potential is associated, as there may be multiple potentials over the same argument each attached to a different object. S_i^j is associated with one or more agent interfaces. For each interface $\hat{I}_i^m$, agent A^j maintains its potential using a data structure called a *linkage tree* which we denote by L_i^m. We denote this potential by $B_{L_i^m}(\hat{I}_i^m)$.

Let LF_i denote the LJF compiled from MSBN M_i, both over the enlarged domain $\hat{V}_i = \cup_j \hat{V}_i^j$. The *joint system potential* (JSP) of LF_i is defined as

$$B_{LF_i}(\hat{V}_i) = \prod_j B_{T_i^j}(\hat{V}_i^j) / \prod_m B_{L_i^m}(\hat{I}_i^m),$$

where $B_{L_i^m}(\hat{I}_i^m)$ is the belief over the mth agent interface held by any relevant agent.

We say that two potentials over the same set of variables are *equivalent* if they differ by no more than a constant factor. A JT T_i^j is *consistent* if for each pair of clusters Q and Q', the potential over $Q \cap Q'$ computed from either cluster is equivalent. After **UnifyBelief** is performed in T_i^j, it is consistent. The LJF LF_i is *locally consistent* if each local JT is consistent.

When subnets S_i^j and S_i^q are related by interface $\hat{I}_i^m$, agent A^j maintains JT T_i^j and linkage tree L_i^{mj}, while A^q maintains T_i^q and L_i^{mq}. JTs T_i^j and T_i^q are *interface consistent* if each of them is consistent and the following potentials are equivalent: the potential over $\hat{I}_i^m$ computed from $B_{T_i^j}(\hat{V}_i^j)$, the potential $B_{L_i^m j}(\hat{I}_i^m)$, the potential over $\hat{I}_i^m$ computed from $B_{T_i^q}(\hat{V}_i^q)$, and the potential $B_{L_i^m q}(\hat{I}_i^m)$. Note that the first two are maintained by A^j and the last two by

[2]Strictly speaking, it also involves potentials over intersections of adjacent clusters, called separators.

A^q. The LJF LF_i is *interface consistent* if every two local JTs related by an interface are interface consistent.

The LJF LF_i is *globally consistent* if it is locally consistent and interface consistent. After **CommunicateBelief** is performed in LF_i, it is globally consistent. The following theorem (based on Section 8.6 in [15]) establishes that **CommunicateBelief** guarantees exact inference in an MSBN.

Theorem 2. *Let the JPD over a domain V be $P(V)$. Let a LJF over V be globally consistent, and its JSP be equivalent to $P(V)$. Let a set obs of observations be entered into the LJF and* **CommunicateBelief** *be performed. Then the following conditions hold:*

1. *The JSP is equivalent to $P(V|obs)$.*
2. *The potential of each local JT over V^j is equivalent to $\sum_{V\setminus V_j} P(V^j|obs)$.*
3. *For each cluster C in each local JT, the cluster potential $B_C(C)$ is equivalent to $\sum_{V\setminus C} P(V|obs)$.*

With the above formal notions on MSBNs, we prove Theorem 1 below.

Proof of Theorem 1: We prove by induction on time interval i. For the base case $i = 0$, we consider execution of **InitialObservation** and the first 6 lines of **Forecast**.

During **InitialObservation**, the LJF loaded in line 1 (denote it by LF_0) is globally consistent and its JSP is equivalent to $P(V_0)$. In line 2, observations are entered at each agent. As agent interfaces are d-sepsets and the interface observability holds, each (enlarged) subdomain $\hat{V}_0^j$ is conditionally independent of each other subdomain $\hat{V}_0^k$ where $k \neq j$, given observations on agent interfaces between them. Therefore, line 3 is equivalent to **CommunicateBelief** without actual communication. After line 3, LF_0 is globally consistent. From Theorem 2 and the fact that FI_0^j is contained in a single cluster in T_0^j, $B(FI_0^j)$ retrieved from that cluster is exact. That is,

$$B(FI_0^j) = const * P(FI_0^j|obs_0^j) = const * P(FI_0^j|obs_0),$$

where 'const' is a constant factor and obs_0 is the set of observations made at $i = 0$ by all agents. Note that the second equality holds due to the interface observability.

Moving to **Forecast**, based on the above argument, $B(FI_0^j)$ retrieved at line 1 is exact. At line 2, LF_1 is loaded. From Lemma 1, marginalization of $B(Q)$ to FI_0^j is a constant distribution. Therefore, before line 4 is executed, $B(Q) = const * P(Q \setminus FI_0^j|FI_0^j)$, and the potential associated with local JT T_1^j is $B(\hat{V}_1^j) = const * P(\hat{V}_1^j \setminus FI_0^j|FI_0^j)$. After line 4 is executed, the potential over Q becomes

$$B'(Q) = const * P(Q \setminus FI_0^j|FI_0^j) * P(FI_0^j|obs_0^j) = const * P(Q|obs_0^j).$$

This implies that the potential of T_1^j becomes

$$B'(\hat{V}_1^j) = const * P(\hat{V}_1^j \setminus FI_0^j|FI_0^j) * P(FI_0^j|obs_0^j) = const * P(\hat{V}_1^j|obs_0^j).$$

That is, the potential over T_1^j has been conditioned on observation obs_0^j. This, however, renders LF_1 no longer globally consistent. Note that the conditional independence between subdomains discussed above does not hold in LF_1 as its agent interfaces have not yet been observed at this point. Line 5 regains global consistence in LF_1. From Theorem 2, the JSP for LF_1 is equivalent to $P(\hat{V}_1|obs_0)$, the potential of T_1^j is equivalent to $P(\hat{V}_1^j|obs_0)$, and for each cluster C of T_1^j, the cluster potential is equivalent to $P(C|obs_0)$. Hence, the forecast for $i = 1$ at line 6 is exact. This concludes the proof for the base case.

We make the inductive assumption for $i = m$. Assume when line 6 of **Forecast** is executed at the end of interval m, LF_{m+1} is globally consistent and its JSP is equivalent to $P(\hat{V}_{m+1}|obs_0, ..., obs_m)$.

Next we consider the time interval $i = m+1$. This involves the last two lines of **Forecast** executed at the start of the interval and the first 6 lines of **Forecast** executed at the end of the same interval (see Figure 6). Each agent completes lines 7 and 8 with respect to LF_{m+1}. Due to d-sepset agent interfaces and interface observability, each subdomain $\hat{V}_{m+1}^j$ is conditionally independent on each other subdomain $\hat{V}_{m+1}^k$ where $k \neq j$, given observations on agent interfaces between them at times $i = 0, ..., m+1$. Therefore, line 8 is equivalent to **CommunicateBelief**. After line 8, LF_{m+1} is globally consistent, and $B(FI_{m+1}^j)$ retrieved from a cluster in T_{m+1}^j in line 1 during the next execution of **Forecast** satisfies

$$B(FI_{m+1}^j) = const * P(FI_{m+1}^j|obs_0^j, ..., obs_{m+1}^j).$$

At line 2, LF_{m+2} is loaded by agents. After line 4, the potential over T_{m+2}^j becomes

$$B'(\hat{V}_{m+2}^j) = const * P(\hat{V}_{m+2}^j|obs_0^j, ..., obs_{m+1}^j),$$

and LF_{m+2} is not globally consistent. After line 5, LF_{m+2} regains global consistence and its JSP is equivalent to $P(\hat{V}_{m+2}|obs_0, ..., obs_{m+1})$. For each cluster C in any local JT, its potential is equivalent to $P(C|obs_0, ..., obs_{m+1})$. Hence, forecast at line 6 on $i = m+2$ is exact. $\square$

Next, we consider the time complexity for one-step-ahead forecasting using **Forecast**. This is essentially the complexity of **CommunicateBelief** which dominates the computation. As **CommunicateBelief** is performed using the LJF, we use the following parameters to characterize the DMSBN and LJF. Let n be the total number of agents in the multiagent DMSBN, κ be the maximum number of possible values of a variable, m be the maximum number of clusters in a local JT, and q be the cardinality of the largest cluster in all local JTs. Then from [15], the time complexity of **CommunicateBelief**, and hence that of **Forecast**, is $O(n\ m\ \kappa^q)$. Since the LJF is reused for each time i, the space complexity of **Forecast** is also $O(n\ m\ \kappa^q)$.

Each subnet S_i^j of DMSBN contains two forward interfaces FI_{i-1}^j and FI_i^j, each of which is contained in a single cluster in the local JT. One of them is often the largest cluster in the local JT (C_4 in Figure 5 (b)). In sparse DMSBNs, the cardinality of this cluster is often close to $|FI_i^j|$. Furthermore, we have $|FI_i^j| = |F_{i+1}^j|$ (see Def. 10). Hence, q can be approximated by the cardinality of the largest set of temporal arcs,

$$q \approx \phi = \max_{i,j} |F_i^j|.$$

Replacing q with ϕ, we have the time complexity of **Forecast** as $O(n\ m\ \kappa^{\phi})$. Note that this is the complexity of computation performed by all agents.

Two observations can be made: First, the complexity is independent of the time. This is important as the computational cost will not grow as other inference methods have experienced (see [14]).

More importantly, the complexity is *exponentially reduced* from that of exact inference based on an equivalent single-agent DBN. In particular, if the spatial distribution is ignored, a DMSBN becomes a DBN with $F_i = \cup_j F_i^j$ being its set of temporal arcs. Denote $\Phi = \max_i |F_i|$ (the maximization has no effect for structural time-invariant DMSBNs). Assume that exact inference is performed with the single-agent DBN using a method such as [7, 8]. Also assume that the number of clusters in the resultant JT has about the same number of clusters as the number of clusters in all local JTs in a LJF. That is, the number of clusters in the resultant JT is approximately $n\ m$. Then the time complexity of exact inference with the single-agent DBN is $O(n\ m\ \kappa^{\Phi})$.

If we assume that the cardinality of F_i^j in the DMSBN does not vary significantly with j, we have the approximate relation $\Phi = n\ \phi$. It then follows that forecasting using the multiagent DMSBN reduces the time complexity exponentially by a factor of

$$\kappa^{\Phi - \phi} = \kappa^{\phi\ (n-1)}.$$

8. Domains Expressible by Parametric Time-Variant DMSBNs

In this section, we consider forecasting in domains expressible by structural time-invariant but parametric time-variant DMSBNs. In such domains, the set of dependence relations (that we care to model) is invariant over time, but CPTs for some variables may drift over time (due to factors that we choose not to represent explicitly). In our supply chain example, suppose that the availability of qualified workers at time i mainly depends on the availability at time $i-1$ and we have chosen to ignore all other factors. However, the distribution $P(worker_i^j | worker_{i-1}^j)$ may drift between two CPTs $\theta(worker_i^j | worker_{i-1}^j)$ and $\psi(worker_i^j | worker_{i-1}^j)$ due to the combined influence from the status of the economy, the season, and so on. With the parametric time variability, it no longer appears feasible to precompile and reuse the same LJF template as in **Forecast**. We show below that is not necessarily the case.

One way to handle the parametric time variability is to add a binary parent variable, say $a_i^j \in \{a^-, a^+\}$, to $worker_i^j$. The CPT associated with $worker_i^j$ is redefined as

$$P(worker_i^j | worker_{i-1}^j, a_i^j = a^-) = \theta(worker_i^j | worker_{i-1}^j),$$
$$P(worker_i^j | worker_{i-1}^j, a_i^j = a^+) = \psi(worker_i^j | worker_{i-1}^j).$$

The drifting of the original distribution is encoded by a temporal arc between a_i^j and a_{i-1}^j, as shown in Figure 7.

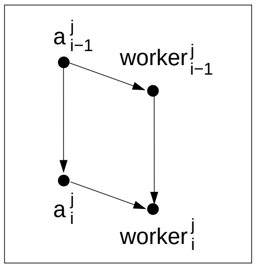

Figure 7: The modified substructure of a parametric time-variant DMSBN.

In general, let the distribution $P(v_i|\pi(v_i))$ be drifting between a set of $w > 1$ CPTs

$$\{\theta[1](v_i|\pi(v_i), ..., \theta[w](v_i|\pi(v_i)\}.$$

Create a new parent variable $a_i \in \{a[1], ..., a[w]\}$ for v_i. Define the new CPT associated with v_i as

$$P(v_i|\pi(v_i), a_i = a[m]) = \theta[m](v_i|\pi(v_i))$$

for $m = 1, ..., w$. Connect a_i and a_{i-1} by a temporal arc and define the CPT $P(a_i|a_{i-1})$. Note that $P(a_i|a_{i-1})$ is independent of i. That is, it is time-invariant.

With the above modification for each relevant variable, the parametric time-variant DMSBN is transformed into a time-invariant DMSBN, and the forecasting algorithm **Forecast** is applicable. Hence, our multiagent forecasting method is applicable to all structural time-invariant DMSBNs no matter whether they are parametric time-invariant or not.

Note that the variable a_i is generally unobservable, but its value can be monitored and predicted as a side effect of **Forecast**.

9. Experiments

We first present our experiments on dynamic domains that can be modeled directly as time-invariant DMSBNs and then on domains that are directly expressible as structural time-invariant and parametric time-variant DMSBNs.

9.1. Domains Expressible by Time-Invariant DMSBNs

To evaluate multiagent forecasting with time-invariant DMSBNs, we compare their forecasting performance with equivalent DBNs. The dynamic domain is a supply chain consisting of a product supplier, a component supplier and a material supplier. Each supplier is aided by an agent. A multiagent team consists of three agents which populate the supply chain DMSBN in Figure 4. A single-agent team consists of three independent agents (no communication and cooperation between them) each of which embodies a different DBN induced by the above DMSBN (see Proposition 1). Hence, the two teams of agents have the same background knowledge about the domain (and will be associated with the same time series) but differ in whether the knowledge is used cooperatively.

The dynamic domain is simulated by logic sampling performed using a centralized DBN that is obtained by ignoring the spatial distribution of the above DMSBN. Each simulation generates a sufficient, multivariate time series of horizon $k = 6$, which we refer to as a *scenario*. The scenario is hidden from both agent teams but are partially revealed to them during forecasting as time i progresses. Therefore, the scenario is consistent with the model carried by agents. Both agents teams and the domain simulation are implemented using the WebWeavr toolkit[3].

Each agent in each team makes one-step-ahead forecast in its subdomain for each of the 6 time intervals. For all three agents in each team, a subtotal of 13 variables (underlined in Figure 4) are forecasted at each step and a total of 78 variables are forecasted over 6 intervals. For each variable, the forecast is its value with the highest posterior marginal and is compared against its value in the scenario. The fraction of variables forecasted correctly by three agents over 6 intervals is used as the performance measure of the agent team, referred to as forecasting accuracy (in the range $[0, 1]$).

Each batch of experiments is conducted on a group of 30 scenarios. For each scenario, five forecasting sessions (S_1,...,S_5) may be run. S_2, S_4 and S_5 are run by the DMSBN agent team, and S_1 and S_3 are run by the DBN agent team. In sessions S_1 and S_2, only variables in agent interfaces are observed as assumed by interface observability. Additional observations are made in S_3 and S_4 (identically). In S_5, both the agent interface and forward interface are observed. Since future events are independent of all current events given the forward interface, agents have the most informative observations in S_5. In short, sessions differ by the agent team and the amount of observations available to agents, which is summarized in Table 1.

Table 1: Forecasting sessions

	S_1	S_2	S_3	S_4	S_5
Agent team	DBN	DMSBN	DBN	DMSBN	DMSBN
Agent int. observed	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
Addition observations			$\surd$	$\surd$	
Forward int. observed					$\surd$

In general, probabilistic reasoning involves inference of both deductive and abductive nature. Consider a directed path $x \rightarrow ... \rightarrow y \rightarrow ... \rightarrow z$ in a BN. If

[3]Available at http://www.cis.uoguelph.ca/$\sim$yxiang/.

the posterior on y is needed, then the observation of x drives deductive inference and the observation of z drives abductive inference. Intuitively, forecasting is similar to deductive inference in the temporal direction, without the assistance of the abductive counterpart. As a result, the accuracy of forecasting is heavily dependent on the causal strength[4] between the current state and future events. To take this dependency into account in our evaluation, we control the level of causal strength of the dynamic domain in the experiment (and parameterize the graphical model accordingly) as follows.

Let v be a variable in the DMSBN associated with $P(v|\pi(v))$. For each instantiation $\underline{\pi(v)}$ of $\pi(v)$, denote $\omega(v|\underline{\pi(v)}) = max_v P(v|\underline{\pi(v)})$, where maximization is over all possible values of v. Hence, $\omega(v|\underline{\pi(v)})$ is a simple indicator of the causal strength. The closer it is to 1, the more predicable the value of v given $\underline{\pi(v)}$. To set the level of causal strength for a DMSBN, a parameter $t \in (0.5, 1)$ is specified, and for each variable v, $\omega(v|\underline{\pi(v)})$ is lower-bounded by t. We simulated three groups of scenarios (30 each), G_1, G_2 and G_3, with strength parameter $t = 0.93, 0.8, 0.7$, respectively.

Figure 8 summarizes the forecasting accuracy for the batch of experiments run on G_1 with causal strength 0.93. By comparing results between S_1 and S_2, and between S_3 and S_4, it can be seen that the DMSBN team has more accurate forecasting than the DBN team. By comparing results between S_1 and S_3, and between S_2, S_4 and S_5, it can be seen that more observations result in more accurate forecasts by each team. The same general trend can be seen in Figure 9 which summarizes the forecasting accuracy for the batch of experiments run on G_2 with causal strength 0.80. Furthermore, S_5 is run on G_3.

Note that each line in the above figures is intended to highlight results from the same team. Since the experiment for each scenario is independent, the slopes of each line have no meaningful relevance.

The statistical significance of the above outcome is evaluated using the student's t-test, and the result is summarized in Table 2. The null hypothesis is that μ_i and μ_j from sessions S_i and S_j are the same. The alternative hypothesis is stated in the 2nd column. In three tests, the alternative hypothesis is accepted with a significance level greater than 99.99%, and in one greater than 96%.

Table 2: Summary of p-values for t-test wrt time-invariant domains.

Sessions compared	Alternative hypothesis	Batch = G_1, causal strength = 0.93	Batch = G_2, causal strength = 0.80
S_1 vs S_2	$\mu_2 > \mu_1$	0.000089608	0.031166752
S_3 vs S_4	$\mu_4 > \mu_3$	0.000000005	0.000177155

In addition, we run session S_5 for each scenario in G_3 with causal strength

[4]We use the term 'causal' loosely here.

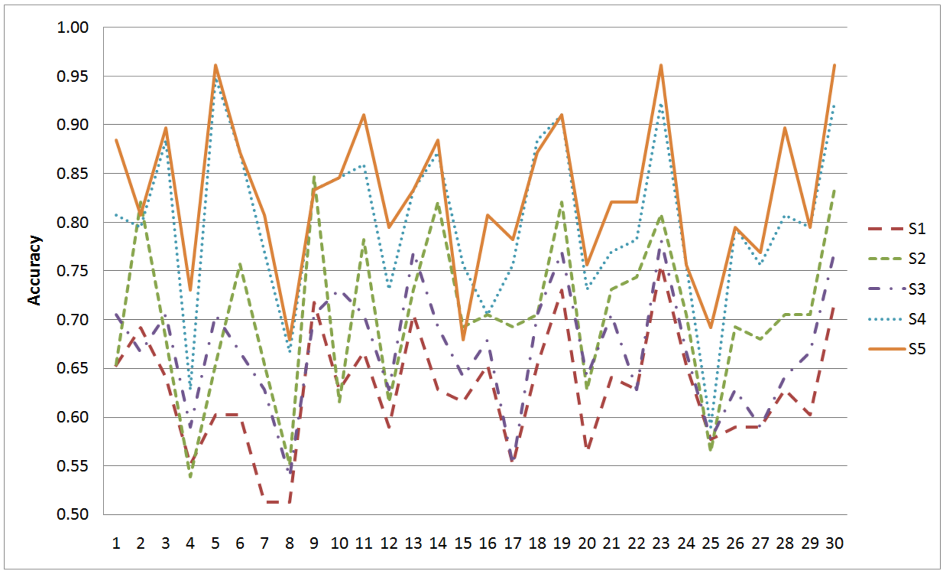

Figure 8: Forecast accuracy for causal strength 0.93. The horizontal axis is labeled by scenario index. The vertical axis is labeled by forecasting accuracy $\in [0, 1]$. Mean accuracies and standard deviations for sessions S_1 through S_5 are as follows: $\mu_1 = 0.63, \sigma_1 = 0.17, \mu_2 = 0.70, \sigma_2 = 0.20, \mu_3 = 0.67, \sigma_3 = 0.17, \mu_4 = 0.80, \sigma_4 = 0.19, \mu_5 = 0.83, \sigma_5 = 0.18$.

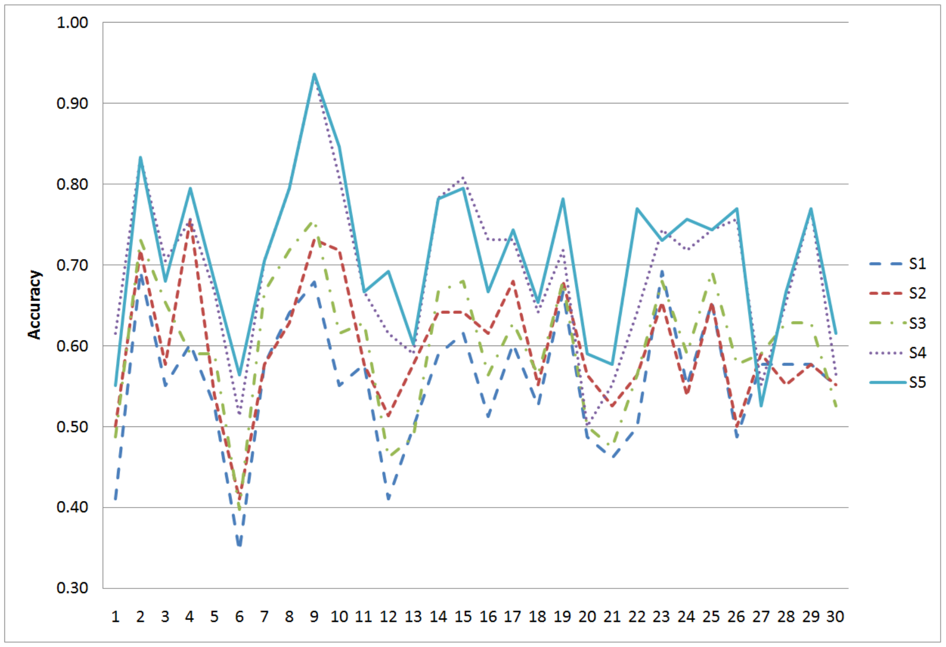

Figure 9: Forecast accuracy for causal strength 0.80. Mean accuracies and standard deviations for S_1 through S_5 are as follows: $\mu_1 = 0.55, \sigma_1 = 0.18, \mu_2 = 0.59, \sigma_2 = 0.20, \mu_3 = 0.60, \sigma_3 = 0.17, \mu_4 = 0.69, \sigma_4 = 0.20, \mu_5 = 0.71, \sigma_5 = 0.20$.

0.70. The average accuracy over the 30 scenarios is $\mu_5 = 0.58$ (standard deviation $\sigma_5 = 0.21$). From the mean accuracies of S_5 in G_1, G_2 and G_3, i.e., 0.83, 0.70 and 0.58, respectively, it is clear that, in dynamic domains of stronger causal strength, more accurate forecasting can be achieved.

9.2. Domains Expressible by Parametric Time-Variant DMSBNs

Next, we present our experiment on domains that are directly expressible by structural time-invariant but parametric time-variant DMSBNs. The parametric time variability of the domain is enabled by making 6 variables in V_i (evenly distributed among three agents) parametrically time-variant. That is, their CPTs are set to drift with time. For three of them, the CPT drifts between two alternative versions and for the other three, the CPT drifts between four alternatives. For each variable above, its CPT will drift with probability 0.07. We refer to these variables as being *parametrically time-variant* or simply time-variant.

We encode the domain using the technique described in Section 8. Denote the corresponding DMSBN as DM_{fv}. An equivalent DBN is obtained from DM_{fv} by ignoring the spatial distribution, and the DBN is used to simulate the dynamic domain. Hence, the simulated domain is directly expressible by a parametric time-variant DMSBN. From the domain, a group G_4 of 30 scenarios are simulated with the strength parameter t = 0.93.

We created three multiagent teams each associated with a different DMSBN. In addition to DM_{fv}, two alternative DMSBNs are created. One of them, DM_{iv}, is a time-invariant DMSBN that does not model the parametric time variability of the domain. In particular, for each time-variant variable, a permanent CPT (identical to one of the versions used in DM_{fv}) is used in DM_{iv}. Hence, agents associated with DM_{iv} ignore the parametric time variability of the domain completely. In the third DMSBN, DM_{hv}, half of the time-variant variables are modeled as in DM_{fv} and the other half as in DM_{iv}. Hence, agents associated with DM_{hv} partially ignore the parametric time variability of the domain.

For each scenario in G_4, three sessions S_6, S_7 and S_8 are run by multiagent teams associated with DM_{iv}, DM_{hv} and DM_{fv}, respectively. The forecasting accuracy for this batch of experiments is summarized in Figure 10. The DM_{fv} team outperformed the other two teams, and the DM_{hv} team outperformed the DM_{iv} team. This demonstrates the effectiveness of encoding parametrical time variability using the technique in Section 8.

Table 3: Summary of p-values for t-test wrt parametric time-variant domains.

Sessions	Alt. hyp.	p-value
S_6 vs S_7	$\mu_7 > \mu_6$	0.000000543
S_7 vs S_8	$\mu_8 > \mu_7$	0.000000001

The statistical significance of the outcome is evaluated using the student's

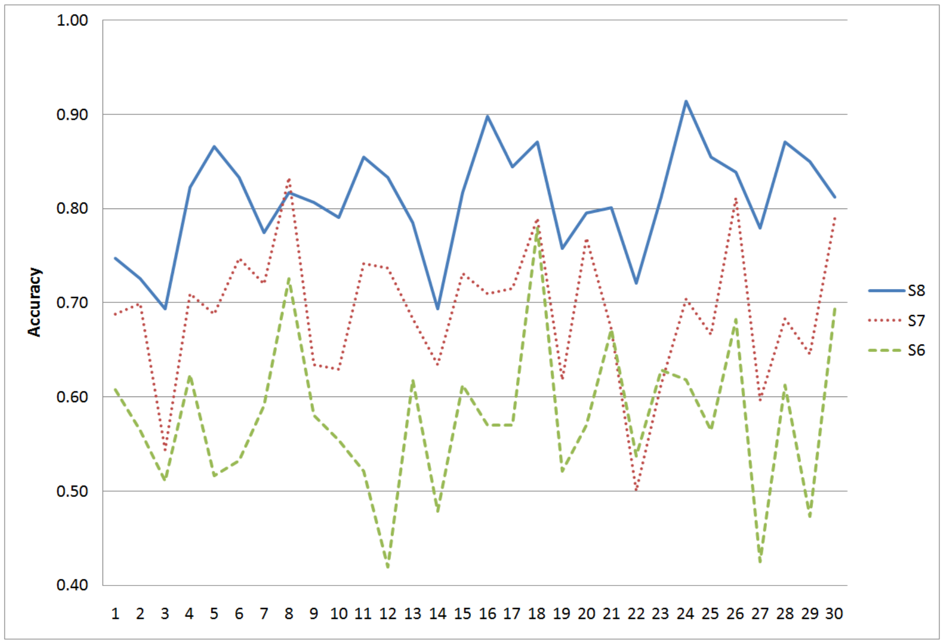

Figure 10: Forecast accuracy for parametrically time-variant domain. Mean accuracies for S_6 through S_8 are as follows: $\mu_6 = 0.58, \mu_7 = 0.69, \mu_8 = 0.81$. Their standard deviations are about 0.13.

t-test, with the result summarized in Table 3. In both tests, the alternative hypothesis is accepted with a significance level greater than 99.99%.

10. Conclusion

Forecasting in stochastic, dynamic domains based on multivariate time series has a wide range of applications and can be performed based on DBNs. A main obstacle in exact forecasting using DBNs is the temporal disintegration of factorization. In [10], a concise closing comment (p564) was made on this difficulty after presenting exact inference in DBNs:

> "The DBN model itself, which represents the prior joint distribution over all the variables, is factorable into its constituent CPTs, but the posterior joint distribution conditioned on an observation sequence - that is, the forward message - is generally not factorable. So far, no one has found a way around this problem, despite the fact that many important areas of science and engineering would benefit enormously from its solution. Thus, we must fall back on approximate methods."

In this contribution, we present a solution to this problem, focusing on a subclass of inference tasks, exact forecasting. We present the multiagent DMSBN framework, whose key components are a distributed, temporal graphical model, a cooperative multiagent organization, the structural time invariability and interface observability assumptions, a compiled runtime representation and a distributed forecasting algorithm suite. The framework employs the runtime LJF which keeps the "forward message" factorized and hence neither the space nor the time complexity increases with time. The key enabling factor is the interface observability. We have shown that, as the result, the time complexity of exact forecasting is reduced exponentially compared with that based on DBNs. This makes exact forecasting feasible in large dynamic domains that are out of reach if restricted to DBN modeling.

Constrained by the temporal disintegration of factorization of DBNs, one could manage the computational cost by splitting a large dynamic domain into several smaller ones and performing exact forecasting in each based on an independent DBN. Similarly with this approach, the DMSBN framework splits the domain spatially into subdomains and encodes each by a DBN (Proposition 1). Contrary to the above approach, the DBNs in the DMSBN framework are tightly coupled and cooperative. As the result, the DMSBN approach produces more accurate forecasting than that based on independent DBNs, as we have demonstrated in experiments. A decision-theoretic comparison of tightly coupled multiagent frameworks versus loosely coupled alternatives, e.g., [17, 18], can be found in [19].

We also presented a simple technique to transform parametric time-variant DMSNs into time-invariant DMSBNs. This allows the computational framework mentioned above to be applicable to any structural time-invariant DMSBNs.

This contribution opens the gate for a number of extensions. Although our focus is on forecasting, the DMSBN framework is applicable to the inference task of monitoring/filtering. Its extension to smoothing is perceivable and needs further investigation.

Although the DMSBN framework is presented under the multiagent paradigm, it can be adapted to single-agent, parallel computation. The representational constraints of DMSBNs will provide guidelines on how to decompose the parallel components effectively.

We have chosen the supply chain domain as the experimental testbed. The advantages are that it is small, intuitive, and comprehensible, which are essential for the first of such experiments. Using this testbed, our experiments have focused on demonstrating the exactness, the effectiveness of stable LJF runtime representation, and improvement on forecasting accuracy by the proposed multiagent framework. Future experiments in much larger domains are needed to demonstrate empirically the computational savings of the framework, and to explore additional efficiency gains that are sanctioned by the general DMSBN framework.

Acknowledgements

We acknowledge the financial support from Discovery Grant, NSERC, Canada, to the first author, and funding from EPSRC CRiSM Initiative through the second author. We thank anonymous reviewers for their helpful comments.

References

[1] P. Brockwell, R. Davis, Time Series: Theory and Methods (2nd Ed.), Springer, New York, 1991.

[2] N. Chan, Time Series: Applications to Finance, John Wiley & Sons, 2002.

[3] A. Pole, M. West, P. Harrison, Applied Bayesian Forecasting and Time Series Analysis, Chapman & Hall, 1994.

[4] R. Dahlhaus, M. Eichler, Causality and graphical models for time series, in: P. Green, N. Hjort, S. Richardson (Eds.), Highly Structured Stochastic Systems, Oxford University Press, 2003, pp. 115–137.

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

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

[7] U. Kjaerulff, A computational scheme for reasoning in dynamic probabilistic networks, in: D. Dubois, M. Wellman, B. D'Ambrosio, P. Smets (Eds.), Proc. 8th Conf. on Uncertainty in Artificial Intelligence, Stanford, CA, 1992, pp. 121–129.

[8] Y. Xiang, Temporally invariant junction tree for inference in dynamic Bayesian network, in: R. Mercer, E. Neufeld (Eds.), Advances in Artificial Intelligence, Springer, 1998, pp. 363–377.

[9] D. Koller, U. Lerner, Sampling in factored dynamic systems, in: A. Doucet, J. de Freitas, N. Gordon (Eds.), Sequential Monte Carlo in Practice, Springer-Verlag, 2001, pp. 445–464.

[10] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 2003.

[11] C. Queen, J. Smith, Multi-regression dynamic models, J. Roy. Statist. Soc. Ser. B 55 (4) (1993) 849–870.

[12] S. Raudys, I. Zliobaite, The multi-agent system for prediction of financial time series, in: Artificial Intelligence and Soft Computing - ICAISC 2006, LNCS 4029, Springer, 2006, pp. 653–662.

[13] C. Kiekintveld, J. Miller, P. Jordan, M. Wellman, Forecasting market prices in a supply chain game, in: Proc. 6th inter. joint conference on autonomous agents and multiagent systems, ACM, 2007, pp. 1–8.

[14] X. An, Y. Xiang, N. Cercone, Dynamic multiagent probabilistic inference, International Journal of Approximate Reasoning 48 (1) (2008) 185–213.

[15] Y. Xiang, Probabilistic Reasoning in Multiagent Systems: A Graphical Models Approach, Cambridge University Press, Cambridge, UK, 2002.

[16] I. Gihman, A. Skorohod, The Theory of Stochastic Processes II, Springer-Verlag, 1975.

[17] P. Gmytrasiewicz, S. Noh, T. Kellogg, Bayesian update of recursive agent models, User Modeling and User-Adapted Interaction 8 (1) (1998) 49–69.

[18] N. Sondberg-Jeppesen, F. Jensen, A pgm framework for recursive modeling of players in simple sequential Bayesian games, Inter. J. Approximate Reasoning 51 (2010) 587–599.

[19] Y. Xiang, F. Hanshar, Comparison of tightly and loosely coupled decision paradigms in multiagent expedition, International Journal of Approximate Reasoning 51 (2010) 600–613.