AALBORG UNIVERSITY

dHugin: A computational system for dynamic time-sliced Bayesian networks

by

Uffe Kjærulff

17 October 1994

To be referenced as:

Kjærulff, U. (1995). dHugin: A computational system for dynamic time-sliced Bayesian networks, *International Journal of Forecasting*, Special Issue on Probability Forecasting. 11:89-111.

INSTITUTE FOR ELECTRONIC SYSTEMS
DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE
Fredrik Bajers Vej 7 — DK 9220 Aalborg — Denmark
Tel.: +45 98 15 85 22 — TELEX 69 790 aub dk

dHugin: A computational system for dynamic time-sliced Bayesian networks

Uffe Kjærulff
Department of Mathematics and Computer Science, Aalborg University
Fredrik Bajers Vej 7E, DK-9220 Aalborg Ø, Denmark
uk@iesd.auc.dk

17 October 1994

Abstract

A computational system for reasoning about dynamic time-sliced systems using Bayesian networks is presented. The system, called dHugin, may be viewed as a generalization of the inference methods of classical discrete time-series analysis in the sense that it allows description of non-linear, discrete multivariate dynamic systems with complex conditional independence structures. The paper introduces the notions of dynamic time-sliced Bayesian networks, a dynamic time window, and common operations on the time window. Inference, pertaining to the time window and time slices preceding it, are formulated in terms of the well-known message passing scheme in junction trees [Jensen et al. (1990)]. Backward smoothing, for example, are performed efficiently through inter-tree message passing. Further, the system provides an efficient Monte-Carlo algorithm for forecasting; i.e., inference pertaining to time slices succeeding the time window. The system has been implemented on top of the Hugin shell [Andersen et al. (1989)].

Key words: Decision support system, expert system, Bayesian belief network, junction tree, recursive graphical model, time series analysis, discrete Markov chain, non-linear multivariate dynamic system, Monte-Carlo algorithm, forecasting

1 Introduction

Over the last decade, decision support systems based upon exact probabilistic inference have attracted an increasing number of researchers of applied artificial intelligence (expert systems) where uncertainty is an intrinsic characteristic of the problem domain; see, for example, Spiegelhalter et al. (1993). Such systems provide models of complex stochastic phenomena through a specification language involving a probability function and a graph which encodes the conditional independence properties of the probability function; that is, the nodes of this independence graph represent domain variables and the links represent dependences among the variables (loosely speaking).

The specification of the, possibly unmanageably high-dimensional, probability function can be broken down into a collection of specifications of manageable local functions (i.e., involving only a few variables) by exploiting these conditional independences. Similarly, inference can be performed by local computations [Lauritzen & Spiegelhalter (1988), Jensen et al. (1990), Shafer & Shenoy (1990), Dawid (1992)].

As an example of such a probabilistic graphical model, consider the problem of estimating the amount of dry matter produced in a specific field of wheat for a specific period of time (e.g. one week). Assume that the 'photo-synthetic activity' of the wheat plants can be expressed through the independent factors 'amount of solar energy' and 'mean temperature' and the dependent factor 'amount of active plant tissue'. Assume further that the latter factor, which shall be termed the 'net leaf area index', can be expressed through the independent factors 'gross leaf area index' and 'percentage of leaf area infected by the mildew fungus'. Finally, the amount of dry matter produced depends on the photo-synthetic activity. Obviously, many other factors must be included if a model of any practical interest is to be constructed, but for purposes of illustration we shall keep it simple.

A graphical representation of the (conditional) dependences and independences among these seven variables can be displayed as in Fig. 1, where an arrow indicate dependence and a missing arrow indicate (conditional) independence. For example, 'Solar energy' and 'Dry matter' are conditionally independent given that the value of 'Photo-synthesis' is known. Later we shall present a method for reading off all independence properties displayed by such a graph.

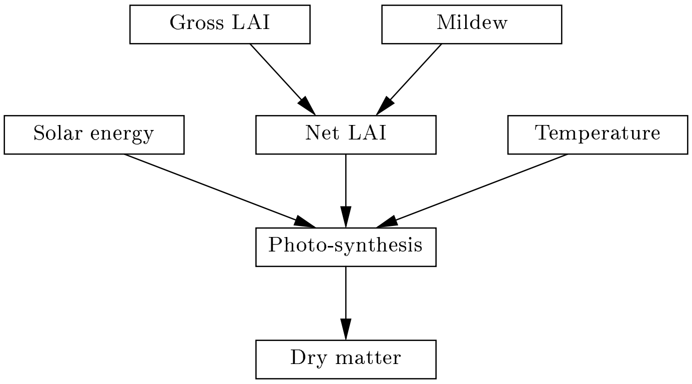

Fig. 1. An over-simplified model of production of dry matter in a wheat field.

The independence graph of Fig. 1 provides the qualitative part of a probabilistic graphical model; henceforth, we shall use the term Bayesian (belief) network for such a model — in the literature, synonyms like causal probabilistic network and recursive graphical model are often used. The quantitative part of the model is given by a probability function defined on the seven variables. The local functions, the product of which comprise the joint distribution p, are given as conditional probability distributions, $p(v \,|\, \mathrm{pa}(v))$, for each variable v given its parents $\mathrm{pa}(v)$ (i.e., the variables from which there are arrows to v); if $\mathrm{pa}(v) = \emptyset$, the local function associated with v reduces to the marginal distribution $p(v)$.

In the small example of Fig. 1, all variables are continuous by nature. However, in the present paper we shall assume all variables to be discrete; continuous variables must be discretized for models like the one in our example to be processed by dHugin.

If the network under consideration is singly connected (i.e., a polytree with at most one path between any pair of nodes), the complexity of exact probabilistic inference is linear in the number of nodes (variables); the graph of Fig. 1 is singly connected. Unfortunately, almost all networks of practical interest are multiply connected, and the general problem of exact inference in multiply connected networks is NP-hard [Cooper (1990)]; that is, we should not expect to find an exact computational method for arbitrary Bayesian networks.

Nevertheless, exact methods have been successfully applied to solve a number of extremely challenging real-world problems. The most well-known examples include the MUNIN network for diagnosing disorders in the peripheral nervous system which contains more than 1000 nodes [Andreassen et al. (1987)], a reconstruction of the QMR/INTERNIST system [Miller et al. (1982)] as a Bayesian network involving 4500 nodes and over 40000 links [Swhe et al. (1991)], and the PATHFINDER network for diagnosing lymph node pathology concerning over 60 diseases and including 109 nodes altogether [Heckerman et al. (1992)].

A common trait of the majority of successful applications of the Bayesian-network paradigm is the static nature of the problem domains. That is, each observable quantity is observed once and for all, and confidence in the observations remaining true is not questioned. However, domains involving repeated observations of a collection of random quantities arise in many fields of science (e.g. medical, economic, biological). For such domains a static model is not very useful: the estimation of probability distributions of domain variables based on appropriate prior knowledge and observation of other domain variables is reliable only for a limited period of time, and further, upon arrival of new observations, both these and the old observations must be taken into account in the reasoning process. Thus, to cope with such dynamic systems using Bayesian networks we need to interconnect multiple instances of static networks, and, as time evolves, add new 'time slices' to the model and cut off old ones. This introduces the notion dynamic time-sliced Bayesian networks (DBNs).

The usefulness of the model of Fig. 1 is rather limited partly because it is over-simplified, but, more importantly, because it will only be applicable for a limited period of time, since it is based on the assumption that a reliable estimate of the amount of dry matter produced can be expressed through mean values of the three determining variables, 'Solar energy', 'Net LAI', and 'Temperature'. Due to the obvious non-linear relationship involved, this assumption holds true only for periods of time of at most, say, a few days or one week. Thus, to provide reliable estimates on gross yield (amount of dry matter at harvest time), a more fine-grained model is required.

What we need is a time-sliced model covering the period from sowing to harvesting with each time slice covering, e.g., one week, and with 'Dry matter (i)' depending on 'Dry matter $(i-1)$' and 'Photo-synthesis (i)', 'Net LAI (i)' depending on 'Net LAI $(i-1)$' and 'Mildew (i)', and 'Mildew (i)' depending on 'Mildew $(i-1)$', the 'micro-climatic' conditions of time period $i-1$, and whether or not spraying against mildew has been performed in time period $i-1$. Such a time-sliced model is indicated in Fig. 2, where time slice n resembles (essentially) the model of Fig. 1 except that 'Mildew (n)' depends on conditions provided at time $n-1$, and time slices $0, \ldots, n-1$ are similar to time slice n except for variables 'Fungicide' (amount of fungicide sprayed on the crop), 'Micro climate', and 'Precipitation'. Notice that the introduction of the 'Micro climate' variable makes the model suitable for forecasting the development of the mildew fungus, and hence comprises possible useful decision support.

If the period from sowing to harvesting covers five months and each time slice in Fig. 2 covers one week, we could, in principle, create an ordinary (i.e., static) Bayesian network including approximately 20 time slices, since our sample application represents a finite process. There are, however, several reasons why this approach would be inexpedient. First, and most importantly, due to the complexity of the model, exact computations will almost surely be prohibitive, depending, of course, on the number of levels (or states) of the (discretized) variables. Second, a straightforward model specification would contain numerous duplicates, since both the qualitative and the quantitative descriptions of the time slices are (almost) identical. Third, the focus of attention is often limited to a few time slices; therefore, the inference procedure should also be focused.

A more efficient and intelligent way of addressing the model specification, the model compilation, and the inference processes would be as follows. (The notion of model compilation

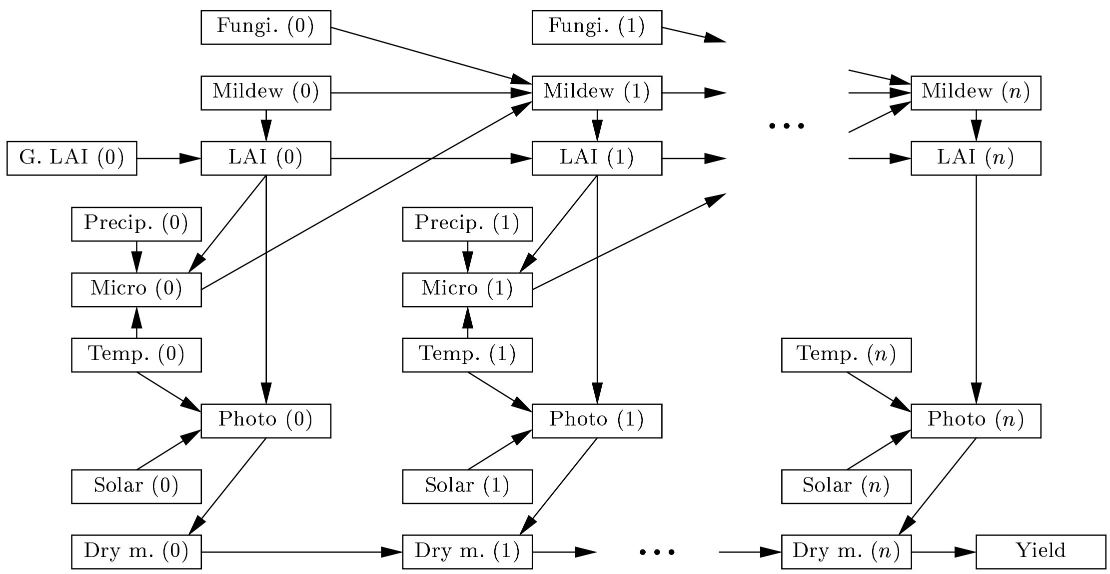

Fig. 2. A sample time-sliced Bayesian network for forecasting the extension of the mildew fungus and estimating/forecasting the gross yield from a field of wheat based on climatic data, observations of leaf area index (LAI) and extension of mildew, and knowledge of amount fungicides used and time of usage.

denotes the process of transforming the model into a tree structure for inference purposes; see Section 2.2.)

Specification of the wheat model in the DBN framework reduces to specifications of the initial and the terminal time slices (i.e., time slices 0 and n) and a specification of time slice 1 (time slices $2, \ldots, n-1$ are identical to time slice 1); the specifications of time slices 1 and n includes specifications of temporal links (i.e., inter-slice links). The actual construction of the dynamic model is then handled by dHugin which 'glues' the relevant time slices together.

Making estimates in a dynamic environment in a way that makes full use of the information about past observations requires a compact representation of this information. The creation of this representation is part of the process of reducing a dynamic model, which includes elimination of parts of the model representing past time slices in such a way that future estimates remain unaffected. That is, the information conveyed by the eliminated part of the model should be completely represented in the resulting model. The complementary process of expanding the model must be carried out whenever new time slices have to be included in the model. This dynamic model shall be denoted the *(time) window*. Typically, the window comprises a small number of time slices. The introduction of the notion of a time window addresses the issue of excessive complexity and the focus-of-attention issue in an elegant way.

In Section 2 we briefly review some relevant graph theoretic concepts as well as some fundamental characteristics of Bayesian networks; furthermore, a precise definition of DBNs and some terminology regarding DBNs are presented. Sections 3–5 provide detailed accounts of the processes of reducing and expanding DBNs, backward and forward propagation, and forecasting. Section 6 discusses issues of possible optimization and improvement of the dHugin system. Section 7 provides a brief account of central related work, and Section 8 concludes the paper by summarizing the features of the dHugin system and mentions some applications of DBNs.

2 Terminology

2.1 Graphs and hypergraphs

A (finite) graph $\mathcal{G} = (V, E)$ consists of a finite set of nodes (vertices) V and a set of links (edges) $E \subseteq V \times V$ of ordered pairs of distinct nodes. If $(v, u) \in E$ and $(u, v) \notin E$ the link from v to u is *directed*. If both $(v, u) \in E$ and $(u, v) \in E$ the link between v and u is *undirected* in which case we shall use the shorthand notation $\{v, u\} \in E$. A graph is said to be directed if all links are directed, and undirected if all links are undirected. Undirectedness shall be understood unless directedness is explicitly assumed.

A (finite) *hypergraph* $\mathcal{H} = (V, \mathcal{C})$ consists of a finite set of nodes V and a set of links $\mathcal{C}$, where $C \subseteq V$ and $|C| \geq 2$ for each $C \in \mathcal{C}$. A hypergraph is a graph if, and only if, each link has cardinality two. $\mathcal{C}$ is called a *hypergraph cover* of a graph $\mathcal{G} = (V, E)$ if there is a $C \in \mathcal{C}$ for each $e \in E$ such that $e \in C$. The graph $G(\mathcal{C})$ of a hypergraph has node set $\bigcup_{C \in \mathcal{C}} C$ and link set $\{\{v, u\} \,|\, \exists C \in \mathcal{C} : v, u \in C\}$.

Let $(\mathcal{G}_1 = (V_1, E_1), \ldots, \mathcal{G}_k = (V_k, E_k))$ be a total ordering of any k graphs such that $V_i \cap V_{i+1} \neq \emptyset$ for all $i = 1, \ldots, k-1$. Then the *compound graph* of $\mathcal{G}_1, \ldots, \mathcal{G}_k$ is given by

$$\bigcup_{i=1}^{k} \mathcal{G}_i = \left(\bigcup_{i=1}^{k} V_i, \bigcup_{i=1}^{k} E_i \right).$$

If there is a link between v and u, denoted by $v \sim u$, v and u are said to be *adjacent* or *neighbours*. The set of neighbours of v is denoted $\mathrm{adj}(v)$, and the set of links between v and $\mathrm{adj}(v)$ are called the links *incident* to v. If there is a directed link from v to u, then v is called a *parent* of u and u a *child* of v. The sets of parents and children of v are denoted $\mathrm{pa}(v)$ and $\mathrm{ch}(v)$, respectively. For a directed graph $\mathcal{G} = (V, E)$, $\mathcal{G}^m$ denotes its *moral graph* obtained by adding undirected links between pairs of nodes with common children and dropping the directions of the links.

A *path* $\pi = \langle v = v_1, \ldots, v_k = u \rangle$ from v to u of length $k-1$ is an ordered sequence of distinct nodes such that $v_i \sim v_{i+1}$ for all $i = 1, \ldots, k-1$. If $v_i \in \mathrm{pa}(v_{i+1})$ for all $i = 1, \ldots, k-1$, then π is a *directed path*. In a directed graph, $\mathcal{G} = (V, E)$, the set $\mathrm{de}(v)$ for which there is a directed path from v to u for each $u \in \mathrm{de}(v)$ is called the *descendants* of v. The set $\mathrm{an}(v)$ for which there is a directed path from u to v in $\mathcal{G}$ for each $u \in \mathrm{an}(v)$ is called the *ancestors* of v. A subset $A \subseteq V$ is an *ancestral set* if $\mathrm{an}(v) \subseteq A$ for all $v \in A$. By $\mathrm{An}(A)$ we denote the smallest ancestral set containing A.

An m-cycle is a path $\langle v = v_1, \ldots, v_k = u \rangle$ of length m with the exception that $v = u$. A *chord* in an m-cycle is a link between v_i and v_j such that $1 < |i - j| < m - 1$ (i.e., it connects the two non-consecutive nodes v_i and v_j of the cycle). A directed graph with no directed cycles is called a *directed, acyclic graph* or DAG for short.

For $A, B, C \subseteq V$, C is said to *separate* A *from* B if for each path $\langle v = v_1, \ldots, v_k = u \rangle$, where $v \in A$ and $u \in B$, $\{v_1, \ldots, v_k\} \cap C \neq \emptyset$. A graph $\mathcal{G}$ is *connected* if there is a path between each pair of nodes of $\mathcal{G}$. Unless otherwise stated, connectivity shall be assumed throughout the paper.

A subset $A \subseteq V$ induces a *subgraph* $\mathcal{G}_A = (A, E_A)$ of $\mathcal{G} = (V, E)$, where $E_A = E \cap (A \times A)$. A graph is *complete* if all nodes are pairwise adjacent. A subset $A \subseteq V$ is complete if it induces a complete subgraph, and if A is maximal (i.e., there is no complete subset $B \subseteq V$ such that $A \subset B$), then it is called a *clique*. A hypergraph $\mathcal{H} = (V, \mathcal{C})$ is *conformal* if every clique of $G(\mathcal{C})$ is contained in a link of $\mathcal{H}$. Thus for any conformal hypergraph with link set $\mathcal{C}$, $\mathcal{C}$ is the set of cliques of $G(\mathcal{C})$.

A *(proper) decomposition* of an undirected graph $\mathcal{G} = (V, E)$ is a triple (A, B, C) of non-empty and disjoint subsets of V such that $V = A \cup B \cup C$, C separates A from B, and C is a complete subset of V. A decomposition (A, B, C) *decomposes* $\mathcal{G}$ into subgraphs $\mathcal{G}_{A\cup C}$ and $\mathcal{G}_{B\cup C}$. $\mathcal{G}$ is *decomposable* if, and only if, (A, B, C) decomposes $\mathcal{G}$ and both $\mathcal{G}_{A\cup C}$ and $\mathcal{G}_{B\cup C}$ are decomposable. $\mathcal{G}$ is *triangulated* (or *chordal*) if each cycle of length greater than 3 has a chord. A graph is triangulated if, and only if, it is decomposable [Lauritzen et al. (1984)].

When a node $v \in V$ and the links incident to v are removed from $\mathcal{G} = (V, E)$, v is said to be *deleted*, but when $\mathrm{adj}(v)$ are made a complete subset by adding the necessary links (if any) to the graph before v and the links incident to v are removed, then v is said to be *eliminated*. Note that connectivity of a graph is invariant under elimination, but not necessarily under deletion. The set, say T, of links added by eliminating all nodes in V in any order is called a *triangulation* of $\mathcal{G}$ as $(V, E\cup T)$ is triangulated. If there is no proper subset $T' \subset T$ such that T' is a triangulation of $\mathcal{G}$, then T is said to be a *minimal triangulation*. The links of T are called *fill-ins*. An *elimination ordering* is a bijection $\# : V \leftrightarrow \{1, \ldots, |V|\}$. $\mathcal{G}_\#$ is an *ordered graph*. The triangulation $T(\mathcal{G}_\#)$ is the set of links produced by eliminating the nodes of $\mathcal{G}$ in order $\#$. An elimination ordering $\#$ is *perfect* if $T(\mathcal{G}_\#) = \emptyset$. The triangulated graph $(V, E \cup T(\mathcal{G}_\#))$ shall be denoted $\mathcal{G}^{t\#}$; for convenience, the same notation shall be used if $\mathcal{G}$ is a DAG (i.e., moralization is understood).

A hypergraph $\mathcal{H} = (V, \mathcal{C})$ is decomposable if, and only if, $G(\mathcal{C})$ is decomposable. Any decomposable hypergraph is conformal [Lauritzen et al. (1984)]. If $\mathcal{C}$ is a cover of some graph $\mathcal{G} = (V, E)$ and $\mathcal{H}$ is decomposable, then $\mathcal{C}$ is called a *decomposable cover* of $\mathcal{G}$. $\mathcal{C}$ is a *minimal decomposable cover* of $\mathcal{G}$ if there is a minimal triangulation, T, of $\mathcal{G}$ such that $\mathcal{C}$ is the clique set of $(V, E \cup T)$.

2.2 Bayesian networks and Markov properties

A Bayesian network, as used in the present paper, is built on a DAG, $\mathcal{G} = (V, E)$, where each node $v \in V$ corresponds to a discrete random variable X_v with finite state space $\mathcal{X}_v$. For $A \subseteq V$, X_A denotes the vector of variables indexed by A. Similarly, $x_A = (x_v)_{v\in A}$ denotes an element of the joint state space $\mathcal{X}_A = \times_{v\in A}\mathcal{X}_v$. Each random variable X_v of a Bayesian network is described in terms of a conditional probability distribution $p(x_v \,|\, x_{\mathrm{pa}(v)})$ defined on $\mathcal{X}_v$, where $p(x_v \,|\, x_{\mathrm{pa}(v)})$ reduces to an unconditional distribution if $\mathrm{pa}(v) = \emptyset$. In $\mathcal{G}$, the conditioning variables of X_v are represented by $\mathrm{pa}(v)$. The joint probability, $p = p_V$, defined on $\mathcal{X}_V$ *factorizes (recursively)* according to $\mathcal{G}$, since

$$p(x) = \prod_{v\in V} p(x_v \,|\, x_{\mathrm{pa}(v)}). \tag{1}$$

$\mathcal{G}$ is called the *independence graph* of p, since it captures all (conditional) independence properties of p. That is, statements of conditional independence between pairs of sets $X_A, X_B \subseteq X_V$ of variables given a third set X_C. We shall use the notation $A \perp\!\!\!\perp B \,|\, C$ for any triple (A, B, C) of disjoint subsets of V for which X_A and X_B are conditionally independent given X_C with respect to p.

All conditional independence statements captured by $\mathcal{G}$ can be found using the d-separation criterion of Pearl (1988) or the equivalent criterion of Lauritzen et al. (1990) expressed by the following theorem.

Theorem 1 [Lauritzen et al. (1990)] *Let p factorize according to a DAG, $\mathcal{G} = (V, E)$. Then $A \perp\!\!\!\perp B \,|\, C$ with respect to p for any subsets $A, B, C \subseteq V$ whenever C separates A from B in $(\mathcal{G}_{An(A\cup B\cup C)})^m$.*

See Lauritzen et al. (1990) for a proof. The property described in Theorem 1 is referred to as the *global Markov property*. We also say that p is Markov with respect to $\mathcal{G}$ when p factorizes according to $\mathcal{G}$.

If a probability function p is Markov with respect to a DAG, $\mathcal{G} = (V, E)$, then there exists a *potential representation*

$$p(x) = z^{-1}\phi(x) = z^{-1} \prod_{C \in \mathcal{C}} \phi_C(x_C), \tag{2}$$

of p, where ϕ_C is a non-negative function for each C in a set $\mathcal{C}$ of complete subsets of V; see, for example, Lauritzen & Spiegelhalter (1988). The ϕ_C's are called *belief potentials*, and z a *normalization constant*. In particular, the product in Equation (1) is a potential representation with normalization constant 1.

2.3 Junction trees

By exploiting the conditional independence relations among the variables of a Bayesian network, the underlying joint probability space may be decomposed into a set of subspaces corresponding to a decomposable (hypergraph) cover of the moralized graph such that exact inference can be performed through a simple message-passing scheme in a maximal spanning tree of the cover [Spiegelhalter (1986), Lauritzen & Spiegelhalter (1988), Jensen (1988), Jensen et al. (1990)]. Technically, a decomposable cover of a Bayesian network with underlying DAG $\mathcal{G}$ is created by triangulating $\mathcal{G}^m$ (i.e., adding undirected links, so-called *fill-ins*, to $\mathcal{G}^m$ to make it triangulated). That is, the set of cliques of the triangulated graph is a decomposable cover of the network.

Jensen (1988) has shown that any maximal spanning tree of a decomposable cover, $\mathcal{C}$, can be used as the basis for a simple inward/outward (or collect/distribute) message-passing scheme for propagation of evidence (belief updating) in Bayesian networks, where maximality is defined in terms of the sum of cardinalities of the intersections between adjacent nodes in the tree. Jensen (1988) named these trees *junction trees*. The nodes of a junction tree are called *(belief) universes* and corresponds to the cliques of the associated triangulated graph; the intersections between neighbouring universes of a junction tree are called *separators* [Jensen et al. (1990)].

We shall henceforth refer to a junction tree by the pair $(\mathcal{C}, \mathcal{S})$ of universes and separators. It can be shown that for each path $\langle C = C_1, \ldots, C_k = D \rangle$ in a junction tree, $C \cap D \subseteq C_i$ for all $1 \leq i \leq k$, which implies that $A \perp\!\!\!\perp B \,|\, S$ for each separator S, where A and B are the sets of variables of the two subtrees (except S) induced by the removal of the link corresponding to S [Jensen (1988)].

To each universe and each separator is associated a (belief) potential, ϕ_A. The joint probability distribution, p, of a Bayesian network with a corresponding junction tree $(\mathcal{C}, \mathcal{S})$ is proportional to the *joint (system) belief* $\phi = \phi_V$ given by

$$p(x) \propto \phi(x) = \frac{\prod_{C \in \mathcal{C}} \phi_C(x_C)}{\prod_{S \in \mathcal{S}} \phi_S(x_S)}. \tag{3}$$

A potential ϕ_A is *normalized* if $\sum_A \phi_A = 1$. If all potentials of a junction tree are normalized, then ϕ_V is normalized (i.e., $p_V = \phi_V$).

The basic operation in passing a message from a universe $C \in \mathcal{C}$ to one of its neighbours $D \in \mathcal{C}$, is referred to as *(belief) absorption*. Let S be the separator associated with C and D. Then D is said to *absorb* from C if the potentials ϕ_D and ϕ_S are changed to ϕ^*_D and ϕ^*_S as

$$\phi^*_S = \sum_{C \setminus D} \phi_C, \quad \text{and} \quad \phi^*_D = \phi_D * \frac{\phi^*_S}{\phi_S}.$$

A junction tree $\Upsilon = (\mathcal{C}, \mathcal{S})$ is said to be *balanced* if

$$\sum_{C \setminus D} \phi_C \propto \sum_{D \setminus C} \phi_D \quad \text{for all } C, D \in \mathcal{C}$$

(i.e., the marginal potentials for $C \cap D$ with respect to ϕ_C and ϕ_D are proportional). Balance of Υ shall interchangeably be referred to as balance of its associated joint belief, ϕ_V. An unbalanced junction tree will become balanced if the operations *CollectEvidence* and *DistributeEvidence* are executed (in that order). *CollectEvidence* takes as argument a universe, say C, and let C absorb from all of its neighbours once they have completed *CollectEvidence* calls to their neighbours (except C). That is, evidence are collected from leaf universes to an arbitrary root universe. *DistributeEvidence* takes as argument a universe C and let all of C's neighbours (except the one from which it was called) absorb from C and then it calls *DistributeEvidence* for each of the neighbours. That is, evidence are distributed from a root universe (the one from which *CollectEvidence* was initially called) to the leaf universes. See Jensen et al. (1990) for details.

Two junction trees $\Upsilon_1 = (\mathcal{C}_1, \mathcal{S}_1)$ and $\Upsilon_2 = (\mathcal{C}_2, \mathcal{S}_2)$ with non-empty and complete intersection $S = C_1 \cap C_2$, where $C_1 \in \mathcal{C}_1$ and $C_2 \in \mathcal{C}_2$, are said to be *consistent* if both Υ_1 and Υ_2 are balanced and $\sum_{C_1 \setminus S} \phi_{C_1} \propto \sum_{C_2 \setminus S} \phi_{C_2}$. If both Υ_1 and Υ_2 are balanced, but they fail to be consistent, then they are said to be *inconsistent*.

2.4 *Dynamic time-sliced Bayesian networks*

At any point in time, a dynamic time-sliced Bayesian network (DBN) covers a (possibly varying) number, say n, of time slices. Let $\mathcal{G} = (V, E)$ be the DAG describing the 'structure' of the dynamic model at a particular point in time. Note that $\mathcal{G}$ is not necessarily an independence graph of the model when the initial time slice (i.e., time slice 0) is not represented explicitly in the model. If t' is the first time slice of the network, then V consists of disjoint subsets, $V(t'), \ldots, V(t'+n-1)$. That is,

$$V = V(t', n) = \bigcup_{t=t'}^{t'+n-1} V(t).$$

The time slices of a DBN are assumed to be chosen such that the DBN obeys the Markov property: the future is conditionally independent of the past given the present. (The notion of 'Markov property' should not be confused with the previously defined notion of 'global Markov property'.) Formally this may be written as

$$X_{V(0)}, \ldots, X_{V(t-1)} \perp\!\!\!\perp X_{V(t+1)}, \ldots, X_{V(t+k)} \,|\, X_{V(t)}$$

for all $t > 0$ and $k > 0$.

The set of directed links

$$E^{\mathrm{tmp}}(t) = \{(v, u) \in E \,|\, v \in V(t-1), u \in V(t)\}, \quad t' \leq t \leq t' + n - 1,$$

is called the *temporal links* (or *temporal relations*) of time slice t and define how the distributions of the variables of time slice t are given conditionally on the distributions of the variables of time slice $t-1$. Thus, temporal links are those between nodes of adjacent time slices in the DAG of a DBN; see the example in Fig. 3.

The subset $\mathrm{int}(t) \subseteq V(t)$ is called the *interface* of time slice t and is defined as

$$\mathrm{int}(t) = \{u \in V(t) \,|\, (v, u) \in E^{\mathrm{tmp}}(t) \text{ or } \exists w \in \mathrm{ch}(u) : (v, w) \in E^{\mathrm{tmp}}(t),\ v \in V(t-1)\}.$$

The moral graph corresponding to the independence graph of the sample DBN in Fig. 3 appears in Fig. 4, where the interfaces are indicated by filled circles. Note that $\mathrm{int}(0) = \emptyset$.

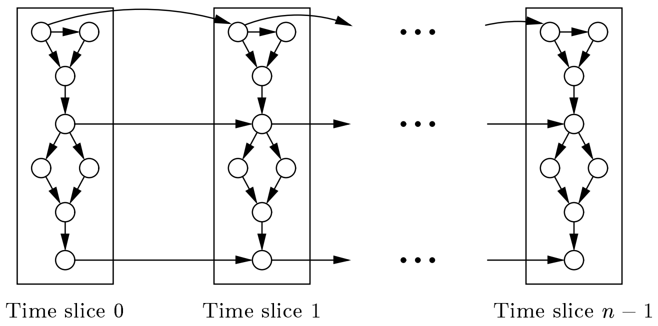

Fig. 3. Independence graph of a sample DBN.

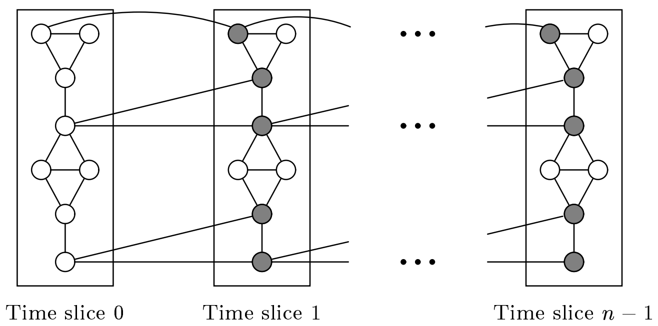

Fig. 4. Moral graph corresponding to the independence graph of Fig. 3, where the interfaces are indicated by filled circles.

The set E of links of $\mathcal{G}$ can now be described as

$$E = E(t', n) = E(t') \cup \bigcup_{t=t'}^{t'+n-1} E^*(t),$$

where $E(t) \subseteq V(t) \otimes V(t)$ and $E^*(t) = E(t) \cup E^{\mathrm{tmp}}(t)$; the symbol $\otimes$ denotes the binary operator producing the set of all ordered pairs of distinct elements of its arguments.

At any point in time, a DBN consists of a series $\mathcal{P}_1, \ldots, \mathcal{P}_{N+1}$ of distinct, but strongly related, submodels. Each $\mathcal{P}_n$ $(1 \le n \le N+1)$ is specified by the quadruple $(p_n, \mathcal{G}_n, t_n, w_n)$, where w_n is the number of time slices represented by $\mathcal{P}_n$, t_n is the oldest time slice represented by $\mathcal{P}_n$, $\mathcal{G}_n = (V_n, E_n)$ is a DAG as defined below, and p_n is a probability function defined on $\mathcal{X}_{V_n}$. $\mathcal{P}_1, \ldots, \mathcal{P}_{N-1}$ are referred to as the *backward smoothing models* and the corresponding time slices $t_1, \ldots, t_{N-1}$ as the *backward smoothing slices*. Thus, $w_n = 1$ whenever $n < N$. The maximal number of backward smoothing slices, which we shall denote by b, is greater than or equal to $N-1$, the actual number of backward smoothing slices. $\mathcal{P}_{N+1}$ is referred to as the *forecast model*, and it comprises $f = w_{N+1}$ time slices $t_{N+1}, \ldots, t_{N+1} + f - 1$ referred to as the *forecast slices*. The submodel $\mathcal{P}_N$ is referred to as the *(time) window*. The window has width $w = w_N$ and thus includes time slices $t_N, \ldots, t_N + w - 1$. See Fig. 5 for a summary.

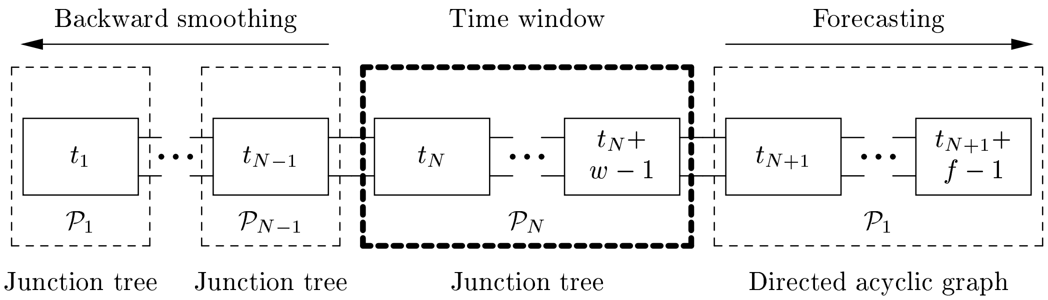

Fig. 5. A DBN consists of a series of submodels $\mathcal{P}_1, \ldots, \mathcal{P}_{N+1}$, where $\mathcal{P}_N$ is called the time window.

Note that the term 'forecast slices' is slightly imprecise since all inference concerning variables of time slices for which no observations (evidence) have been entered, actually are forecasts, even if such time slices belong to the time window. For similar reasons the term 'backward smoothing slices' is also slightly imprecise.

The DAGs $\mathcal{G}_1, \ldots, \mathcal{G}_{N+1}$ are given as follows.

$$\mathcal{G}_n = (V_n, E_n) = \begin{cases} (V(t_n, 1) \cup \mathrm{int}(t_{n+1}), E(t_n, 1) \cup E'(t_{n+1})) & \text{if } n < N \\ (V(t_n, w), E(t_n, w)) & \text{if } n = N \\ (V(t_{N+1}, f) \cup V_N^*, E(t_{N+1}, f) \cup E^{\mathrm{tmp}}(t_{N+1})) & \text{if } n = N+1, \end{cases} \tag{4}$$

where

$$E'(t_{n+1}) = E^*(t_{n+1}) \cup \left((\mathrm{int}(t_{n+1}) \otimes \mathrm{int}(t_{n+1})) \cap E(t_{n+1}) \right),$$
$$V_N^* = \{ v \,|\, (v, \cdot) \in E^{\mathrm{tmp}}(t_{N+1}) \}.$$

Note that for each $n < N$, $\mathcal{G}_n$ contains the interface of slice t_{n+1}, and $\mathcal{G}_{N+1}$ contains the nodes of $\mathcal{G}_N$ from which there are temporal links to nodes of slice t_{N+1}. Fig. 6 shows an example, where $N = 3$ and $w = f = 2$.

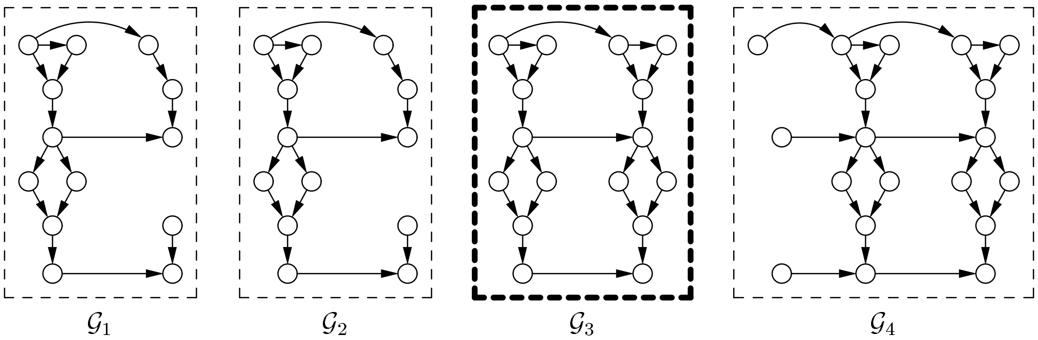

Fig. 6. A sample DBN consisting of 4 submodels with associated DAGs $\mathcal{G}_1, \ldots, \mathcal{G}_4$. The backward smoothing models, $\mathcal{P}_1$ and $\mathcal{P}_2$, contain the interface variables of submodels $\mathcal{P}_2$ and $\mathcal{P}_3$, respectively. The forecast model, $\mathcal{P}_4$, contains variables of the time window needed to make forecasts.

3 Expansion and reduction of DBNs

Based on the terminology established above, we shall now discuss issues of moving the window forward in time. This process involves the two more or less separate processes of *window expansion* and *window reduction* discussed in detail in Sections 3.2 and 3.3. Since the time window is represented by a junction tree, these two processes roughly amount to, respectively, adding a new subtree to the junction tree and cutting off a part of the tree.

Window expansion by, say, k new time slices consists of (a) adding k new consecutive time slices to the forecast model, (b) moving the k oldest time slices of the forecast model to the time window, (c) moralizing the compound graph including the triangulated graph of the window and the DAGs of the k new time slices, (d) triangulating that graph and identifying the new clique set, (e) constructing the new (expanded) junction tree, and (f) computing the new universe/clique potentials with due consideration for the old ones. As discussed in Section 3.2, the last step is optional. Expanding the window by k new time slices causes the width, w, of the window to be increased by k, while the number, f, of forecast slices remains unaltered.

In principle, window reduction by k time slices involves elimination of all variables pertaining to time slices $t_N, \ldots, t_N + k - 1$. (Recall that elimination of a node v forms a complete subset of the nodes $\mathrm{adj}(v)$ unless the set is already complete.) The end-product of the elimination process is a potential involving the variables $\mathrm{int}(t_N + k)$. This potential, represented in one of the universes of the reduced junction tree, Υ_N say, represents all information about the past necessary for the reduced window to take full account of the knowledge about the history of the dynamic system. If $k \leq b - (N-1)$ (i.e., k is less than or equal to the difference between the maximal and the actual number of backward smoothing slices), then N is increased by k; otherwise, it is increased by $b-(N-1)$. In both cases the width, w, of the window is decreased by k, but the number, f, of forecast slices remains unaltered. Note that when the maximal number of backward smoothing slices is decreased by m, say, then N is decreased by $m - (b - (N-1))$ if $m > b - (N-1)$; otherwise, it remains unaltered.

Two issues are of major importance here: (a) if backward smoothing is to be performed, the universes corresponding to the cliques of the triangulated graph resulting from the reduction process must be linked together in a new junction tree, Υ_{N-1} say, such that backward smoothing can be performed by passing messages from Υ_N to Υ_{N-1} via the potential involving variables $\mathrm{int}(t_N)$, and (b) since both the expansion and the reduction processes perform a triangulation (i.e., finds an elimination ordering) of (basically) the same graph, these two processes should be coordinated such that the same elimination ordering is employed.

The triangulation carried out as a subtask of the expansion process is unconstrained in the sense that the search space of elimination orderings consists of all permutations of the set, V, of nodes of the (expanded) window, whereas the reduction process may be perceived as a constrained triangulation, where the eliminated set $A \subset V$ of nodes define the prefix of orderings comprising all nodes in V. Then obviously it might be advantageous to make a constrained triangulation (decomposition) in the first place, rendering the reduction process trivial, provided it is carried out in the fundamental way described above (i.e., assuming the reduction concerns all nodes of the k oldest time slices of the window).

3.1 Constrained elimination orderings

A constrained elimination ordering is defined as follows.

Definition 1 Let $\mathcal{P}_1, \ldots, \mathcal{P}_N$ be a series of submodels of a DBN, $\mathcal{G}^N = (V, E)$ the compound graph of $\mathcal{G}_1 = (V_1, E_1), \ldots, \mathcal{G}_N = (V_N, E_N)$, and $\# : V \leftrightarrow \{1, \ldots, |V|\}$ a bijection defining a total ordering of V. The ordering $\#$ is said to be *constrained* if $\#(v) < \#(u)$ for all $1 \leq i < j \leq N$, where $v \in V_i$ and $u \in V_j$. Similarly, $T((\mathcal{G}^N_{\#})^m)$ is said to be a *constrained triangulation* of $(\mathcal{G}^N)^m$.

Constrained elimination orderings have a number of important properties which shall be used in Sections 3.3 and 4.

Rose et al. (1976) have shown that $v \sim u$ in a triangulated graph $\mathcal{G}^t_{\#}$ if, and only if, all nodes on a path between v and u in $\mathcal{G}_{\#}$ are of lower order than both v and u (i.e., they are eliminated before v and u). We state that property formally by the following lemma.

Lemma 1 [Rose et al. (1976)] *Let $\mathcal{G}_{\#} = (V, E)$ be an ordered graph. Then $v \sim u$ in $\mathcal{G}^t_{\#}$ if, and only if, there is a path $\langle v, v_1, \ldots, v_k, u\rangle$ in $\mathcal{G}_{\#} = (V, E)$ such that $\#(v_i) < \min\{\#(v), \#(u)\}$ for all $i = 1, \ldots, k$.*

See Rose et al. (1976) for a proof.

Lemma 1 can now be used to prove that, under constrained elimination, the interfaces of a DBN induce complete separators of the compound graph of the triangulated graphs corresponding to the DAGs of the DBN.

Lemma 2 *Let $\mathcal{P}_1, \ldots, \mathcal{P}_N$ be a series of submodels of a DBN, $\mathcal{G}^N = (V, E)$ the compound graph of $\mathcal{G}_1, \ldots, \mathcal{G}_N$, and $(\mathcal{G}^N)^*$ the compound graph of the triangulated graphs $\mathcal{G}^{t\#}_1, \ldots, \mathcal{G}^{t\#}_N$, where $\#$ defines a constrained ordering of V. Then int(t) is a complete separator in $(\mathcal{G}^N)^*$ for all $t = t_1, \ldots, t_N + w - 1$.*

Proof: Let n refer to any of the submodels $\mathcal{P}_1, \ldots, \mathcal{P}_N$ and let t refer to any of the time slices $t_n, \ldots, t_n + w_n - 1$. Since $\#$ defines a constrained ordering, nodes in $V(t-1)$ are eliminated before $\mathrm{int}(t)$ in $\mathcal{G}_n$. Hence, by Lemma 1, $v \sim u$ in $\mathcal{G}^{t\#}_n$ for each $v, u \in \mathrm{int}(t)$ (i.e., $\mathrm{int}(t)$ is complete in $\mathcal{G}^{t\#}_n$). Since, by definition, $\mathrm{int}(t)$ is a separator of $(\mathcal{G}^N)^m$ and nodes in $V(t-1)$ are eliminated before $\mathrm{int}(t)$, then $v \not\sim u$ in $(\mathcal{G}^N)^*$ for each $v \in V(t-1)$ and each $u \in V(t) \setminus \mathrm{int}(t)$. Thus $\mathrm{int}(t)$ is a complete separator in $(\mathcal{G}^N)^*$. $\square$

Corollary 1 *Let $\mathcal{P}_1, \ldots, \mathcal{P}_N$ be a series of submodels of a DBN, $\mathcal{G}^N = (V, E)$ the compound graph of the triangulated graphs $\mathcal{G}^{t\#}_1, \ldots, \mathcal{G}^{t\#}_N$ associated with the submodels, where $\#$ defines a constrained ordering of V. Then $\mathcal{G}^N$ is constrainedly triangulated.*

Proof: By Lemma 2, $\mathcal{G}^N$ has complete separators $\mathrm{int}(t_1), \dots, \mathrm{int}(t_N + w - 1)$. Thus, for any $t \in \{t_1, \dots, t_N + w - 1\}$, $(A, B, \mathrm{int}(t))$ is a decomposition of $\mathcal{G}^N$, where $A = V(t_1) \cup \cdots \cup V(t-1)$ and $B = V \setminus (A \cup \mathrm{int}(t))$. The graphs $\mathcal{G}^N_{A \cup \mathrm{int}(t)}$ and $\mathcal{G}^N_{B \cup \mathrm{int}(t)}$ have complete separators $\mathrm{int}(t_1), \dots, \mathrm{int}(t)$ and $\mathrm{int}(t), \dots, \mathrm{int}(t_N + w - 1)$, respectively. Thus, there are decompositions $(\cdot, \cdot, \mathrm{int}(t'))$ and $(\cdot, \cdot, \mathrm{int}(t''))$ for $\mathcal{G}^N_{A \cup \mathrm{int}(t)}$ and $\mathcal{G}^N_{B \cup \mathrm{int}(t)}$, respectively. Continuing this argument we end up with subgraphs $\mathcal{G}_1, \dots, \mathcal{G}_N$ all of which are constrainedly triangulated, and the result follows. □

3.2 Window expansion

The operation of window expansion refers to the processes of adding k new time slices $t_{N+1} + f, \dots, t_{N+1} + f + k$ to the forecast model and subsequent transferral of the k oldest time slices, $t_N + w = t_{N+1}, \dots, t_{N+1} + k - 1$, of the forecast model to the time window. The wish to expand the window may be explicit, or implicit as part of the operation of moving the window k time slices forward.

A new time slice, $t_{N+1} + f$, to be added to the forecast model contains one or more variables for which their associated conditional probability functions are specified in terms of variables belonging to the forecast model. Due to the assumption of compliance with the Markov property, all such variables of the forecast model must be included in time slice $t_{N+1} + f - 1$ (i.e., the last slice of the model). In many applications the time slices added to a dynamic model will be identical. However, apart from compliance with the Markov property, the subnetworks (time slices) added to the model can be specified absolutely freely (i.e., there are no assumptions of the time slices added being identical or similar in some sense, although in many applications this is often the case).

Moving the k oldest time slices of the forecast model to the time window is conceptually straightforward. We simply create the compound graph of the triangulated graph of the time window and the DAG induced by the k oldest time slices of the forecast model, and then add the subset of nodes of the (expanded) time window which have children belonging to the (reduced) forecast model (i.e., the subset referred to as V_N^* above). In order to produce a junction tree for the expanded window we perform the operations of moralization and triangulation. Moralization is performed on the compound graph consisting of both directed and undirected links. After (constrained) triangulation, the potentials (i.e., conditional probabilities) of the new variables are attached to appropriate universes.

A sample window expansion is shown in Fig. 7 (initial model consisting of a time window covering the initial time slice and no forecast slices is not shown). Assume now that we want to expand the window by time slice 1 and let the forecast model comprise a single time slice (time slice 2). First, we add time slices 1 and 2 to the forecast model and, since the forecast model is currently void, dummy variables a, d, and h pertaining to time slice 0 is added to the model to make it self-contained (i.e., such that forecasts, by means of Monte-Carlo simulation, can be done without reference to the time window once appropriate information has been received from the window). Second, time slice 1 is moved from the forecast domain to the window and the resulting graph is moralized (moral links indicated by dashed lines) and triangulated (fill-ins indicated by dotted lines). Finally, the new expanded junction tree is created as described below.

Obviously, in finding an optimal elimination ordering, we have to consider the topology of the graph as it appears after the addition of time slice $t_N + w$ (i.e., the next one to be added). Since we want the model complexity in terms of the size of the state space to be as low as possible to minimize the complexity of inference, and since the size of the state space varies heavily over the range of elimination orderings, a careful analysis must be conducted to establish an appropriate ordering. To find an optimum elimination ordering for an arbitrary graph is,

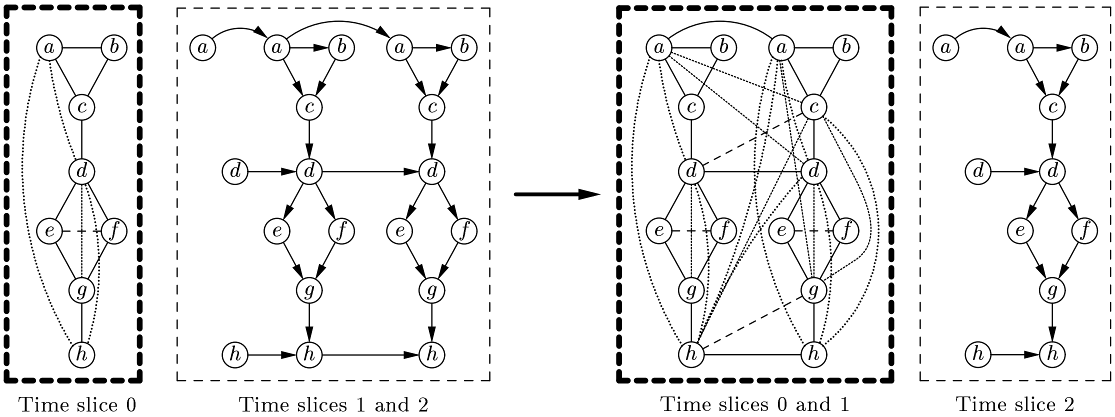

Fig. 7. Sample window expansion. The initial situation (not shown) is given by a DBN model consisting of a time window comprising time slice 0 and a void forecast model. The expansion of both the window and the forecast model by one time slice requires addition of two new time slices, 1 and 2. Time slice 1 is subsequently moved from the forecast model to the time window which is then moralized (dashed links) and triangulated (dotted links).

however, an NP-hard problem [Arnborg et al. (1987)]. Yet, in practice it turns out that near optimum triangulations may be found using simple heuristic ordering strategies [Rose (1973), Kjærulff (1992)]. The elimination ordering employed to obtain the triangulated graph of time slice 0 in Fig. 7 is b, e, f, c, g, d, a, h. (The corresponding DAG and moral graph are shown in Fig. 3 and Fig. 4.)

Having identified the cliques of the new triangulated graph of the window, the next step concerns construction of a corresponding junction tree. As much as possible of the junction tree, $\Upsilon = (\mathcal{C}, \mathcal{S})$, in existence prior to the expansion should be reused in order to minimize the effort required to construct the expanded junction tree $\Upsilon' = (\mathcal{C}', \mathcal{S}')$. Note that as a direct consequence of the constrained triangulation scheme, there is for each 'old' universe $C \in \mathcal{C}$ a 'new' universe $C' \in \mathcal{C}'$ such that $C \subseteq C'$. For some universes the containment might be strict (i.e., they are rendered redundant). The creation of Υ' can be described as follows.

(1) Identify the set $\mathcal{C}'$ of universes of Υ'.

(2) Construct a 'skeleton' of Υ':

  (a) Create a universe for each member of $\mathcal{C}' \setminus \mathcal{C}$ and a separator for each member of $\mathcal{S}' \setminus \mathcal{S}$.

  (b) Create unit-potential tables for these new universes and separators (i.e., tables with each cell instantiated to 1). (The potentials of the universes in $\mathcal{C} \cap \mathcal{C}'$ and of the separators in $\mathcal{S} \cap \mathcal{S}'$ remain unaltered.)

(3) For each member of $\mathcal{C} \setminus \mathcal{C}'$ and each member of $\mathcal{S} \setminus \mathcal{S}'$ (i.e., 'old' universes rendered redundant and the separators incident to them), multiply the associated potentials to the potentials of appropriate universes and separators and attach the result to these objects.

(4) Attach the conditional probability tables of the variables of the new time slices to appropriate new universes.

(The term 'appropriate' in Steps (3) and (4) refers to the index set of the table to be attached being a subset of the set of variables of the universe or separator to which it is attached.)

Fig. 8 illustrates the modifications of the time-window junction tree corresponding to the sample window expansion in Fig. 7, showing the junction tree before and after the expansion. The numbers attached to the universes in Fig. 7 correspond to the order of creation using the above elimination ordering. The old universe 5 becomes redundant, since it is a proper subset of the new universe 5. Thus the potential of the old universe 5 and the potentials of its incident separators are transferred to, respectively, the new universe 5 and relevant separators incident to it (indicated by the dashed arrows). The universes (and separators) remaining unaltered are shown in bold.

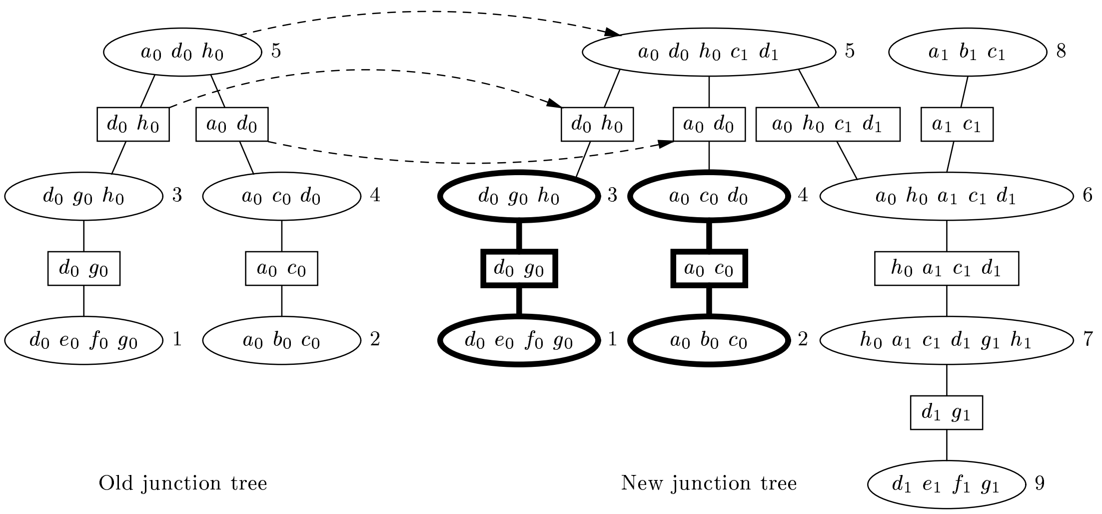

Fig. 8. Modifications of the time-window junction tree corresponding to the window expansion in Fig. 7. Since the old universe 5 becomes redundant, potentials of this universe and its incident separators are transferred to the new junction tree. The parts shown in bold are inherited from the old junction tree.

Now, if we have an immediate interest in the marginal distributions of variables of the k new time slices of the window, propagation can be performed; otherwise we might postpone the propagation step until, for example, new observations arrive. Notice that if Υ was balanced immediately before the window expansion was executed, we only need to perform propagation in the subtree induced by the set of new universes (Fig. 9).

3.3 Window reduction

Due to the constrained decomposition scheme employed by the window expansion process, window reduction becomes relatively easy as previously discussed. In developing a window reduction scheme it is important to recognize the requirements for convenient backward smoothing beyond the time window. Below we develop a reduction scheme which meets such requirements and which is based on the following result.

Theorem 2 *Let $\mathcal{P}_1, \ldots, \mathcal{P}_N$ be a series of balanced submodels, with associated constrainedly decomposable covers. Assume that $\mathcal{P}_{n-1}$ and $\mathcal{P}_n$ are inconsistent for some $1 < n \leq N$. Sufficient*

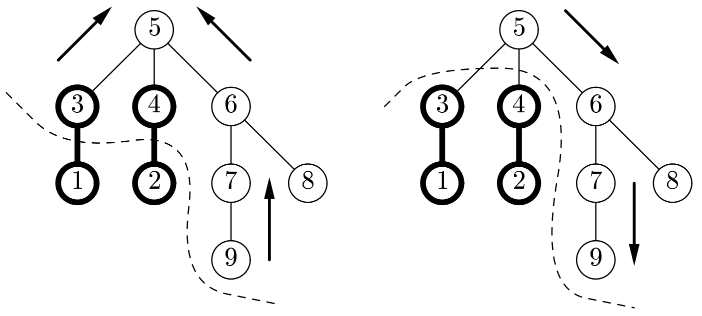

Fig. 9. If the junction tree was balanced before expansion was performed, balance of the expanded junction tree can be obtained through propagation in a subtree: (i) perform *CollectEvidence* in the subtree induced by the new universes *and* the 'old' universes which are adjacent to new universes, and (ii) perform *DistributeEvidence* in the subtree induced by the new universes.

information required to render $\mathcal{P}_{n-1}$ and $\mathcal{P}_n$ consistent is given by the marginal potentials $\phi_{int(t_n)}$, where there is a universe C_1 of $\mathcal{P}_{n-1}$ and a universe C_2 of $\mathcal{P}_n$ such that $int(t_n) \subset C_1$ and $int(t_n) \subseteq C_2$.

Proof: Let $\mathcal{G}^t_{n-1}$ and $\mathcal{G}^t_n$ be the (constrainedly) triangulated graphs corresponding to the decomposable covers of $\mathcal{P}_{n-1}$ and $\mathcal{P}_n$, and let $\#$ define the associated constrained elimination ordering. From Lemma 2 we have that $\mathrm{int}(t_n)$ is a complete separator of $\mathcal{G}^t_{n-1} \cup \mathcal{G}^t_n$ and hence $\phi_{\mathrm{int}(t_n)}$ contains complete mutual information between $\mathcal{P}_{n-1}$ and $\mathcal{P}_n$. From the definition of $\mathcal{G}_i$, $1 \leq i < N$, (cf. Equation (4)) we have that $\mathrm{int}(t_n) \subset V(t_{n-1})$, and since for each pair $\{v, u\}$, where $v \in V(t_{n-1})$ and $u \in \mathrm{int}(t_n)$, $\#(v) < \#(u)$, $\mathrm{int}(t_n)$ induces a complete subgraph of $\mathcal{G}^t_{n-1}$. Hence there is a clique C_1 of $\mathcal{G}^t_{n-1}$ such that $\mathrm{int}(t_n) \subset C_1$. Since $\mathrm{int}(t_n)$ is complete in $\mathcal{G}^t_{n-1}$ it follows immediately that it is also complete in $\mathcal{G}^t_n$ and hence there is a clique C_2 in $\mathcal{G}^t_n$ such that $\mathrm{int}(t_n) \subseteq C_2$. □

Reduction of the window by k time slices partition $\mathcal{P}_N$ into $k+1$ submodels, where the first k submodels become the k consecutive backward smoothing models immediately preceding the new (reduced) time window. The $(k+1)$'st submodel is the new time window. If the new number of backward smoothing slices, $N-1$, exceeds the maximal number, b, then we dispose of the submodels representing the $(N-1)-b$ oldest time slices and let $N = b+1$. That is, whenever $\mathcal{P}_N$ is subjected to a reduction of k time slices, the net increase of N is k if $k \leq b-(N-1)$; otherwise, it is $b-(N-1)$.

In terms of operations on $\mathcal{G}_N$ (i.e., the DAG of the time window), the window reduction process produces $k+1$ new DAGs, $\mathcal{G}_{N-k}, \ldots, \mathcal{G}_N$, complying with Equation (4) and with N updated as described above. In terms of operations on the junction tree of the time window, the reduction process may be formulated as follows.

(1) Let $\Upsilon_0 = (\mathcal{C}_0, \mathcal{S}_0)$ be a balanced junction tree of the time window.

(2) For $i = 0$ to $k-1$ do

   (a) Let $\mathcal{C}'_i = \{C \in \mathcal{C}_i \,|\, C \cap V(t_N + i) \neq \emptyset\}$ be the set of universes containing variables of time slice $T_N + i$. Let $\mathcal{C}''_i = \mathcal{C}_i \setminus \mathcal{C}'_i$ be the remaining universes.

   (b) Let $\Upsilon'_i = \Upsilon_{\mathcal{C}'_i}$ and $\Upsilon''_i = \Upsilon_{\mathcal{C}''_i}$ be the junction trees induced by $\mathcal{C}'_i$ and $\mathcal{C}''_i$, respectively (Fig. 10).

(c) Let $\mathcal{B}_i = \{C \in \mathcal{C}''_i \,|\, \mathrm{adj}(C) \cap \mathcal{C}'_i \neq \emptyset \text{ in } \Upsilon_i\}$ be the set of universes of Υ''_i which are adjacent to universes of Υ'_i.

(d) If there is no $C \in \mathcal{B}_i$ such that $\mathrm{int}(t_N + i + 1) \subseteq C$, then create a universe comprising $\mathrm{int}(t_N + i + 1)$, add it to Υ''_i, and let its neighbours be $\mathcal{B}_i$; otherwise, add $\mathcal{B}_i \setminus \{C\}$ to the neighbour set of C.

(e) Let $\Upsilon_{i+1} = (\mathcal{C}_{i+1}, \mathcal{S}_{i+1}) = \Upsilon''_i = (\mathcal{C}''_i, \mathcal{S}''_i)$.

(3) Let $N := N + k$. If $N - 1 > b$, then dispose of the submodels representing the $(N-1) - b$ oldest time slices and let $N := b + 1$.

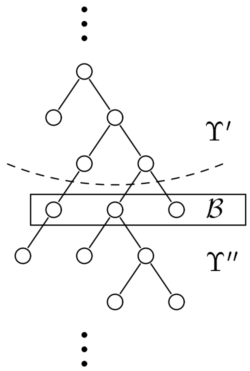

Fig. 10. Partitioning Υ into Υ'_i and Υ''_i.

After the execution of Steps (1)–(3), the junction trees $\Upsilon'_0, \ldots, \Upsilon'_{k-1}, \Upsilon''_{k-1}$ are consistent and correspond to submodels $\mathcal{P}_{N-k}, \ldots, \mathcal{P}_N$, respectively. To see that $\Upsilon'_0, \ldots, \Upsilon'_{k-1}$ are junction trees of $\mathcal{P}_{N-k}, \ldots, \mathcal{P}_{N-1}$ it suffices to realize that any subtree of a junction tree is a junction tree and that $\bigcup_{C \in \mathcal{C}'_i} C$ corresponds exactly to the node set of $\mathcal{G}_i$ for all $i = 0, \ldots, k-2$ (cf. Equation (4)).

The fact that $\Upsilon_k = \Upsilon''_{k-1}$ is a junction tree of the new time window can be seen as follows, where we let $\mathcal{G}^t_N$ be the (constrainedly) triangulated graph corresponding to the decomposable cover of $\mathcal{P}_N$.

First, assume that the condition of the 'if' part of Step (2d) holds. Since the constrained triangulation forces $\mathrm{int}(t_N + k)$ to induce a complete subgraph of $\mathcal{G}^t_N$ and since there is no clique in $\mathcal{C}''_{k-1}$ containing $\mathrm{int}(t_N + k)$, then $\mathrm{int}(t_N + k)$ itself must be a clique of $\mathcal{G}^t_N$. The subset $\mathcal{B}_{k-1} \subseteq \mathcal{C}''_{k-1}$, where for each $B \in \mathcal{B}$ there is a non-empty intersection between the adjacency set of B and $\mathcal{C}'_{k-1}$ in Υ_{k-1}, is then made the adjacency set of the new universe comprising $\mathrm{int}(t_N + k)$. Since the path in Υ_{k-1} between any pair of elements in $\mathcal{B}_{k-1}$ includes elements in $\mathcal{C}'_{k-1}$ (i.e., $\mathcal{C}'_{k-1}$ separates the elements of $\mathcal{B}_{k-1}$ from one another), this does not violate the tree structure of Υ_k. Neither does it violate the property of Υ_k being a junction tree, as the intersection of any pair (C', C'') of universes, where $C' \in \mathcal{C}'_{k-1}$ and $C'' \in \mathcal{C}''_{k-1}$, is a subset of $\mathrm{int}(t_N + k)$.

Second, assume the condition to fail (i.e., there is a universe $C \in \mathcal{C}''_{k-1}$ such that $\mathrm{int}(t_N + k) \subseteq C$) in which case $\mathcal{B}_{k-1} \setminus \{C\}$ is made a subset of the adjacency set of C in Υ_k. With arguments similar to those above it is readily realized that the property of Υ_k being a junction tree is not violated.

4 Forward and backward propagation

The above description of the expansion and reduction processes provides a framework for providing a comprehensible account of issues related to propagation of evidence in DBNs. As indicated in Fig. 5, for each $n \leq N$, inference involving submodel $\mathcal{P}_n$ is carried out in a junction tree representation of p_n, whereas inference in the forecast model, $\mathcal{P}_{N+1}$, is performed in a DAG using Monte-Carlo methods (Section 5). Transferral of information between two consecutive submodels $\mathcal{P}_n$ and $\mathcal{P}_{n+1}$ is allowed only when $n < N$ (i.e., when both submodels are represented as junction trees). Information can be transferred from the time window to the forecast model, but not vice versa. Transferral involving two junction trees is termed *forward propagation* or *backward propagation* depending on the direction of transferral.

Clearly, the arrival of observations (evidence) affects not only the calculation of revised beliefs on variables of the time slice(s) pertaining to the observed variables, but may also have significant impact on beliefs on variables of other time slices. Clearly, this impact goes forward as well as backward in time.

The process of revising beliefs of variables of past slices in light of new evidence (retrospective assessment) is often referred to as *backward smoothing*. If the variables for which revised beliefs are required are all included in the submodel, $\mathcal{P}_n$, for which evidence has been obtained, then backward smoothing is an implicit part of propagation in the junction tree of $\mathcal{P}_n$. If, however, we want to calculate revised beliefs for variables pertaining to submodel $\mathcal{P}_{n-k}$ $(0 < k < n)$, then sufficient information about the observations should be propagated backward from $\mathcal{P}_n$ to $\mathcal{P}_{n-k}$ through belief updating of the submodels $\mathcal{P}_{n-1}, \ldots, \mathcal{P}_{n-k-1}$ (in that order). Similarly, if $n < N$, then appropriate information should be propagated forward from $\mathcal{P}_n$ to $\mathcal{P}_N$ via the interjacent submodels.

From the description of the window reduction scheme (Steps (1)–(3) in Section 3.3) it is clear that for each pair of submodels $\mathcal{P}_n, \mathcal{P}_{n+1}$ $(1 \leq n < N)$ both of the associated junction trees contain a universe including the interface of time slice t_{n+1}, $\mathrm{int}(t_{n+1})$. Let these universes be C and D, respectively. By Theorem 2, the marginal potentials $\sum_{C\setminus D} \phi_C$ and $\sum_{D\setminus C} \phi_D$ on $\mathrm{int}(t_{n+1})$ both provide sufficient information to render $\mathcal{P}_n$ and $\mathcal{P}_{n+1}$ consistent, provided both $\mathcal{P}_n$ and $\mathcal{P}_{n+1}$ are balanced. Therefore, transferral of information from e.g. $\mathcal{P}_n$ to $\mathcal{P}_{n+1}$ is performed through absorption

$$\phi^*_D = \phi_D * \frac{\sum_{C\setminus D} \phi_C}{\sum_{D\setminus C} \phi_D}$$

followed by execution of *DistributeEvidence* in D.

To summarize, each submodel $\mathcal{P}_n$ $(1 \leq n < N)$ has two *interface universes* IC^-_n and IC^+_n, where $\mathrm{int}(t_n) \in IC^-_n$ and $\mathrm{int}(t_{n+1}) \in IC^+_n$, and $\mathcal{P}_N$ has one interface universe $IC^-_N \ni \mathrm{int}(t_N)$. ($IC^-_1$ is undefined if $t_1 = 0$; i.e., if $\mathcal{P}_1$ represents the initial time slice.) When performing forward propagation from $\mathcal{P}_n$ $(1 \leq n < N)$ to $\mathcal{P}_m$ $(n < m \leq N)$, any submodel $\mathcal{P}_i$ $(n < i < m)$ absorbs information from IC^+_{i-1} (through IC^-_n), propagates it, and emits information from IC^+_n; which is absorbed by $\mathcal{P}_{i+1}$ through IC^-_{n+1}, etc. (Fig. 11). The process of backward propagation is similar.

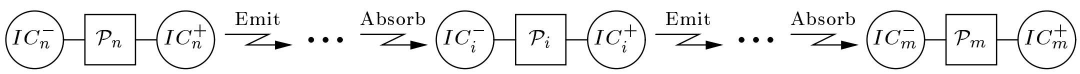

Fig. 11. Forward propagation from $\mathcal{P}_n$ to $\mathcal{P}_m$.

5 Forecasting

In time-series analysis applications there is typically a desire to make forecasts of the random process considered. That is, to calculate estimates of the distributions of future variables given observations of past and present variables.

Within the computational framework presented above, forecasts involving variables of the time window are provided as by-products of propagation within the window. Forecasts involving variables beyond the window could, in principle, be obtained through a two-step procedure involving window expansion and propagation. However, this brute-force approach easily leads to excessive computational complexity if a large number of time slices is added to the window.

A more intelligent way of making forecasts through exact computation would be to advance the time window an appropriate number of steps (i.e., a series of alternating reduction/expansion steps), where propagation is performed in each step, and then move it back to its original position when the desired forecasts have been computed. This method has at least two possible drawbacks: first, it might be a very time consuming operation, and second, it often involves a lot of unnecessary computations. The latter problem stems from the facts that (i) exact computation often forms a glaring contrast to the desired accuracy and to the reliability that, in general, can be attached to forecasts, and (ii) forecasts are typically wanted only for a small number of variables.

Therefore, there is a demand for alternative forecast methods which either avoid the junction-tree approach and/or exploit the fact that forecasts are only required for a small number of variables. We shall address the first of these two issues.

5.1 Forward sampling

The Monte-Carlo sampling scheme denoted 'forward sampling' is an efficient way of calculating single-variable marginal distributions in directed Markov fields when observations are unavailable for variables with non-empty parent sets [Henrion (1988)].

With p being a probability function defined on $\mathcal{X}_V$, where p is Markov with respect to a DAG $\mathcal{G} = (V, E)$, our objective is to provide an estimate of a parameter

$$\theta = E\phi = \sum_{x\in\mathcal{X}} \phi(x)p(x) = \lim_{m\to\infty} \frac{1}{m} \sum_{i=1}^{m} \phi(X_i),$$

where $X_1, \ldots, X_m$ are independent and identically distributed random variables associated with samples from $\mathcal{X}$ with respect to p (i.e., x_i is sampled with probability $p(x_i)$). An unbiased estimate of θ can be obtained as

$$\hat{\theta} = \frac{1}{m} \sum_{i=1}^{m} \phi(X_i). \tag{5}$$

The variance of $\hat{\theta}$ is given by

$$\begin{aligned} \text{Var } \hat{\theta} &= \frac{1}{m}\text{Var } \phi(X) = \frac{1}{m} \sum_{x\in\mathcal{X}} (\phi(x) - \theta)^2 p(x) \\ &\approx \frac{1}{m^2} \sum_{i=1}^{m} (\phi(X_i) - \hat{\theta})^2. \end{aligned} \tag{6}$$

So the standard deviation of $\hat{\theta}$ is proportional to $1/\sqrt{m}$, which implies that a decrease of the standard deviation (or standard error) by a factor of k requires an increase of the sample size by k^2. Thus, to get forecasts within a small distance from the exact values, we should expect the computing time to be relatively large; in some cases even larger than those required by exact methods. On the other hand, the space requirements will obviously be much less, since the sampling is performed in a DAG structure involving relatively low-dimensional probability tables. Further, the time complexity grows only linearly in the dimensionality of the tables involved, whereas it grows exponentially for exact methods.

If p is Markov with respect to a DAG $\mathcal{G} = (V, E)$, a sample x_i from p may be obtained as follows. Let $A \subseteq V$ be the set of orphan nodes of $\mathcal{G}$. Since observations are unavailable, except possibly for members of A, A constitute a set of marginally independent variables. Hence these variables may be instantiated independently through sampling from their marginal distributions. Now, given these instantiations, the orphan nodes of $\mathcal{G}_{V \setminus A}$ are conditionally independent, and may hence be instantiated independently through sampling from their conditional distributions given the samples associated with A. Continuing this process until all variables have been instantiated, we have a sample $x_i \in \mathcal{X}$. Note that there is always at least one orphan node of a DAG, due to the absence of directed cycles. This way of sampling from $\mathcal{X}$ is denoted *forward sampling*.

However, since the orphan nodes of our forecast models are generally dependent due to common ancestors in preceding time slices, a slightly modified sampling technique must be adopted; for example, $a_i \not\perp\!\!\!\perp d_i \,|\, (c_i, c_{i-1})$, $i > 1$, in our sample DBN (Fig. 7). That is, these variables must be instantiated through sampling from their joint distribution. This distribution, however, might not be represented explicitly in the time-window junction tree. Instead of computing this distribution, dHugin employs the 'random propagation' algorithm [Dawid (1992)] by which a sample from the joint distribution associated with a junction tree can be obtained through local sampling in the universes of the junction tree.

Given the above sampling procedure, we may derive estimates of the marginal probabilities of the variables. That is, $\theta = (\theta_v^a = P(X_v = a)), v \in V, a \in \mathcal{X}_v$. In the following two sections we consider two different estimation schemes.

5.1.1 *Empirical estimation*

The empirical estimation scheme is given by

$$\phi(x) = \left(\phi_v^a(x) = \begin{cases} 1 & \text{if } x_v = a \\ 0 & \text{otherwise} \end{cases} \right), v \in V, a \in \mathcal{X}_v,$$

such that $\theta_v^a = E\phi_v^a$, where the expectation is taken with respect to p, and $\hat{\theta}_v^a = (\sum \phi_v^a)/m$. That is, the estimator $\hat{\theta}_v^a$ of $p_v(a)$ is given simply as the relative number of 'counts in cell $(\mathcal{X}_v)_a$' (i.e., the frequency by which state a is selected).

The fact that ϕ is a mapping from $\mathcal{X}$ to $\{0, 1\}$ enables a substantial reduction of Equation (6):

$$\begin{aligned} \operatorname{Var} \hat{\theta}_v^a &\approx \frac{1}{m^2} \sum_{i=1}^{m} \left(\phi_v^a(x_i) - \hat{\theta}_v^a \right)^2 = \frac{1}{m^2} \left[\sum_{i=1}^{n} \left(1 - \hat{\theta}_v^a\right)^2 + \sum_{i=1}^{m-n} \left(\hat{\theta}_v^a\right)^2 \right] \\ &= \frac{1}{m} \left[\frac{n}{m} \left(1 - \hat{\theta}_v^a\right)^2 + \frac{m-n}{m} \left(\hat{\theta}_v^a\right)^2 \right] = \frac{1}{m} \left[\hat{\theta}_v^a \left(1 - \hat{\theta}_v^a\right)^2 + \left(1 - \hat{\theta}_v^a\right) \left(\hat{\theta}_v^a\right)^2 \right] \\ &= \frac{\hat{\theta}_v^a \left(1 - \hat{\theta}_v^a\right)}{m}, \end{aligned} \tag{7}$$

where $n = \sum_{i=1}^{m} \phi_v^a(x_i)$, i.e., the number of times $(x_i)_v = a$.

5.1.2 Mixture estimation

The computation of estimates for the distribution of X_v using empirical estimation is based on samples from $\mathcal{X}_v$ with respect to $p(X_v \,|\, x_{\mathrm{pa}(v)})$. These samples serve two purposes. First, together with samples for the other parents they specify the distribution with respect to which samples from $\mathcal{X}_{\mathrm{ch}(v)}$ should be drawn. Second, they provide the basis for calculating $\hat{\theta}_v^a$, $a \in \mathcal{X}_v$. Intuitively, however, it seems more efficient to calculate an estimate for the distribution of X_v by averaging over the sample densities $p(X_v \,|\, x_{\mathrm{pa}(v)})$, i.e.,

$$\tilde{\theta}_v = \frac{1}{m} \sum_{i=1}^{m} p\left((X_i)_v \,|\, (x_i)_{\mathrm{pa}(v)}\right),$$

The variance of $\tilde{\theta}_v^a$ is given as

$$\begin{aligned}
\mathrm{Var}\ \tilde{\theta}_v^a &= \mathrm{Var}\ p(x_v^a \,|\, X_{\mathrm{pa}(v)}) = \frac{1}{m} \sum_{y \in \mathcal{X}_{\mathrm{pa}(v)}} \left(p(x_v^a \,|\, y) - p(x_v^a)\right)^2 p(y) \\
&= \frac{1}{m} \sum_{y \in \mathcal{X}_{\mathrm{pa}(v)}} p(x_v^a \,|\, y)^2 p(y) + p(x_v^a)^2 p(y) - 2p(x_v^a \,|\, y) p(x_v^a) p(y) \\
&= \frac{1}{m} \left\{ \left[\sum_{y \in \mathcal{X}_{\mathrm{pa}(v)}} p(x_v^a \,|\, y) p(x_v^a, y) \right] - p(x_v^a)^2 \right\}.
\end{aligned}$$

Theorem 3 *Var* $\tilde{\theta} \leq$ *Var* $\hat{\theta}$.

Proof: With $\mathrm{Var}\ \hat{\theta}_v^a = p(x_v^a)(1 - p(x_v^a))/m$ we get

$$\begin{aligned}
\mathrm{Var}\ \tilde{\theta}_v^a - \mathrm{Var}\ \hat{\theta}_v^a &= \frac{1}{m} \left\{ \left[\sum_{y \in \mathcal{X}_{\mathrm{pa}(v)}} p(x_v^a \,|\, y) p(x_v^a, y) \right] - p(x_v^a)^2 - p(x_v^a)(1 - p(x_v^a)) \right\} \\
&= \frac{1}{m} \sum_{y \in \mathcal{X}_{\mathrm{pa}(v)}} p(x_v^a \,|\, y) p(x_v^a, y) - p(x_v^a, y) \\
&= \frac{1}{m} \sum_{y \in \mathcal{X}_{\mathrm{pa}(v)}} \left(p(x_v^a \,|\, y) - 1\right) p(x_v^a, y) \leq 0.
\end{aligned}$$

□

Based on the samples, an approximate expression for $\mathrm{Var}\ \tilde{\theta}_v^a$ is

$$\mathrm{Var}\ \tilde{\theta}_v^a \approx \frac{1}{m^2} \sum_{i=1}^{m} \left(p\left(x_v^a \,|\, (x_i)_{\mathrm{pa}(v)}\right) - \tilde{\theta}_v^a\right)^2. \tag{8}$$

However, calculating $\mathrm{Var}\ \tilde{\theta}_v^a$ using this formula requires an m-dimensional vector of probabilities $p(x_v^a \,|\, (x_i)_{\mathrm{pa}(v)})$ for each state $a \in \mathcal{X}_v$ of each variable $X_v \in X_V$. To circumvent such an excessive space requirement we might instead calculate the difference between $\mathrm{Var}\ \hat{\theta}_v^a$ and $\mathrm{Var}\ \tilde{\theta}_v^a$ and subtract it from $\mathrm{Var}\ \hat{\theta}_v^a$. Hence, based on Equation (8) an approximation of $\mathrm{Var}\ \tilde{\theta}_v^a$ may be obtained as

$$\begin{aligned}
\mathrm{Var}\ \tilde{\theta}_v^a &= \mathrm{Var}\ \hat{\theta}_v^a - \left(\mathrm{Var}\ \hat{\theta}_v^a - \mathrm{Var}\ \tilde{\theta}_v^a\right) \\
&\approx \frac{1}{m} \tilde{\theta}_v^a \left(1 - \tilde{\theta}_v^a\right) - \frac{1}{m^2} \sum_{i=1}^{m} p\left(x_v^a \,|\, (x_i)_{\mathrm{pa}(v)}\right) - p\left(x_v^a \,|\, (x_i)_{\mathrm{pa}(v)}\right)^2, \qquad (9)
\end{aligned}$$

where the sum can be calculated 'on the fly', enabling a reduction in space requirement by a factor of m.

5.1.3 *Comparison of empirical and mixture estimation*

Unfortunately, the ratio between the rates of convergence for the two sampling schemes is not constant, but relies heavily on the characteristics of the distribution, p, from which we sample. The sharper p is, the closer the ratio will be to unity. In the limit where the conditional probabilities of the Bayesian network are purely logical the two schemes are equivalent. If $k = \mathrm{Var}\ \hat{\theta}_v^a / \mathrm{Var}\ \tilde{\theta}_v^a$ with a sample size of m, then mixture estimation provides estimates with a precision similar to that of empirical estimation with only m/k samples. Experiments show that typical k-values range from 2 to 10 [Kjærulff (1993a)]. However, since the computational complexities of the two are identical and since we are not interested in the samples themselves, we shall stick to mixture estimation.

5.1.4 *Confidence interval for* θ_v^a

Given the sample size m, an estimator ξ_v^a of θ_v^a ($v \in V$, $a \in \mathcal{X}_v$), the variance of ξ_v^a, and a confidence level γ (0.95, 0.99 or the like), a confidence interval for θ_v^a can be computed by utilizing the asymptotic behaviour of the distribution function for the estimator.

The following is a standard cook-book recipe for computing the confidence interval for θ_v^a.

(1) Choose a confidence level e.g. $\gamma = 0.95$.

(2) Determine c such that $P(-c \le \theta_v^a \le c) = \gamma$, i.e., such that $\Phi(c) = (1+\gamma)/2$, where Φ is the distribution function for the standard normal density.

(3) Compute $k = c\sqrt{\mathrm{Var}\ \xi_v^a}$. The confidence interval for θ_v^a at level γ is then

$$\xi_v^a - k \le \theta_v^a \le \xi_v^a + k.$$

6 System optimization and improvement

This section discusses some advanced topics related to utilization of (i) special DBN structures for optimizing the efficiency of exact inference and (ii) approximate methods for reduction of computational complexity. None of the suggested methods have been implemented in the current version of dHugin.

6.1 *Utilization of independence relations induced by observations*

The functionality of the current version of dHugin includes the methods described in Sections 3–5. Thus the system does not utilize the possibility for reduction of space complexity imposed by observations. Put differently, observations may induce independence relations such that expansion of the time window may induce fewer fill-ins and hence smaller universes. As demonstrated by the following example, a substantial reduction of space requirements can be obtained.

Consider the sample window expansion in Fig. 7 and assume that evidence on variable h is available at each time slice. When evidence on a variable has been entered and a corresponding belief updating (i.e., propagation) has taken place, the associated node and its incident links may be removed without affecting the correctness of subsequent belief updatings. Therefore, assuming that all variables have identical state-space sizes, an optimal elimination ordering is now h, g, f, e, b, c, d, a. Consequently, the triangulated graph corresponding to the initial time window covering time slice 0, contains only one fill-in (Fig. 12). More importantly, though, the

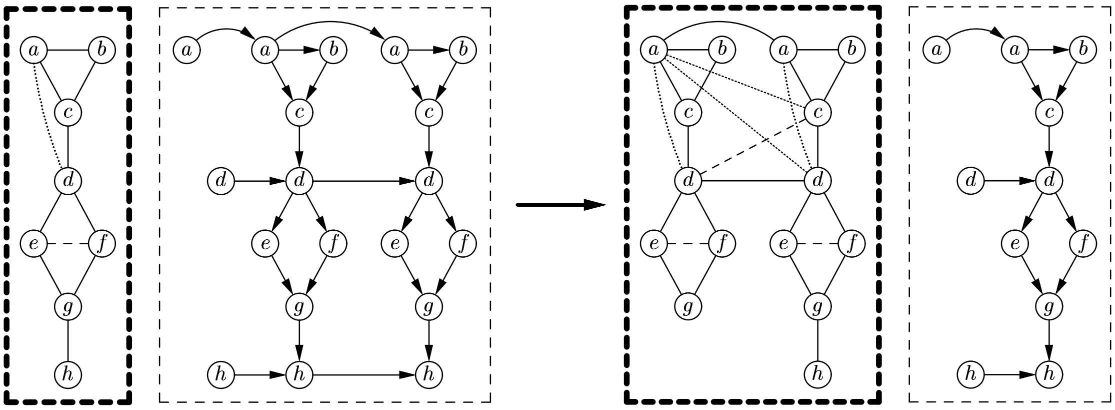

Fig. 12. Utilization of independence relations induced by observing variable h at each time slice enables significant reduction of the space complexity.

removal of the link between g and h implies that the number of fill-ins of the triangulated graph of the expanded time window is reduced from 16 to 4, and the sum of the state-space sizes of the universes is reduced from 192 to 96 if all variables are assumed to be binary. The size of the reduction increases dramatical with an increasing number of possible states per variable. For example, if each variable has 5 possible states, the space requirement is reduced to less than one tenth of the original requirement. Note that, in Fig. 12, variable h of time slice 0 cannot be removed until slice 1 has been added, since we need the associated evidence when calculating the potentials of the expanded window.

A couple of interesting questions naturally arise. What is the criterion for obtaining significant reductions and does the utilization of evidence to reduce space requirements fit into our computational system?

First, if re-triangulation shall be avoided (i.e., finding another (better) triangulation of an already triangulated graph), all evidence available for variables of time slice $t_N + w - 1$ (the newest time slice of the window) must be entered and propagated before time slice $t_N + w$ is added to the window. Thus only one time slice must be added at a time. Further, if $\mathcal{G}$ is the triangulated graph of the time window before expansion takes place, $\mathcal{G}'$ the triangulated graph of the expanded time window, and A the set of variables pertaining to the added time slice, then by Lemma 1 and the constrained elimination scheme, it follows immediately that the complexity of $\mathcal{G}'_A$ is unaltered as long as $\mathcal{G}$ is connected. Thus, significant reductions of the space requirements are achieved only if $\mathcal{G}$ becomes disconnected.

Second, if $\mathcal{G}$ becomes disconnected, it is easily realized that the interfaces split into subsets, one for each connected component of $\mathcal{G}$. If, for example, g is observed instead of h in our example, then the original interface $\{a, c, d, g, h\}$ splits into subsets $\{a, c, d\}$ and $\{g, h\}$. Further, if window reduction is carried out such that the reduced window contains only time slice 1, then forward and backward propagation between the resulting submodels $\mathcal{P}_1$ and $\mathcal{P}_2$ would involve three junction trees $\mathcal{J}_{1a}$, $\mathcal{J}_{1b}$, and $\mathcal{J}_2$ ($\mathcal{P}_1$ consists of two (disconnected) junction trees $\mathcal{J}_{1a}$ and $\mathcal{J}_{1b}$), where $\mathcal{J}_2$ would contain two interface universes, one (containing $\{a, c, d\}$) used for transferral of information between $\mathcal{J}_2$ and $\mathcal{J}_{1a}$ and the other (containing $\{g, h\}$) for transferral of information between $\mathcal{J}_2$ and $\mathcal{J}_{1b}$. Thus, basically, this fits into our computational system. However, to serve as a basis for implementation, a slight revision is necessary.

6.2 Trading off time for reduction of space complexity

Provided a fixed set of variables is guaranteed to be observed at each time slice, the above method provides a simple and effective means of reducing the space complexity. However, if different variables are observed at different time slices or anticipated observations are missing from time to time, this method cannot be used.

In Fig. 7, the triangulated graph associated with the initial model contains four fill-ins even though the moral graph is already triangulated. As discussed in Section 3.2, this apparent waste of space is caused by the elimination ordering b, e, f, c, g, d, a, h which is optimal when considered in the perspective of an expanded window. However, if, for example, variable d is observed at time 0, we would be much better off using the decomposable cover given by the cliques of the moral graph and removing d before expanding the window. On the other hand, if, for example, b and e were observed, then the cover obtained by adding the four fill-ins is preferable with respect to time complexity, but indifferent with respect to space complexity.

Thus, a scheme focusing on minimum-space complexity should use a myopic wait-and-see approach by producing locally optimal covers. The cost of such an approach appears as a possible need to re-triangulate the triangulated graph of the time window before expansion can take place.

6.3 Other issues

For some applications, neither of the approaches discussed in the two preceding sections will be able to provide sufficient reductions of the space complexity. In such situations more drastic solutions must be addressed. One is to employ inference schemes based (partly) on stochastic simulation (i.e., Monte-Carlo methods). It is well-known, however, that the time complexity of such methods may be very high. If we stick to the junction tree approach for inference, we might be forced to accept approximations in the expansion and reduction processes. One obvious, but rather extreme, way to approximate the window expansion process is to assume independence (given evidence) among the parents of variables to be added to the window. An indication of the error hereby made could be calculated by employing, for example, information theoretic metrics providing measures of mutual information among the relevant variables. If we apply this approximate method on the sample window expansion in Fig. 7, then no fill-ins are needed at all, implying a reduction of the space requirement from 192 to 76 for the expanded time window if all variables are binary. If the number of possible states per variable is five, then the space requirement is reduced by more than a factor of 20.

Improved methods for forecasting is another field, where further work could be done. Due to the relatively large number of iterations of the forward sampling schemes that might be required for an acceptable precision, the time complexity of these and other Monte-Carlo methods can be fairly high. One alternative approach could be to apply the approximate window expansion method described above. Since no evidence is available for the variables of the forecast model, the approximation is likely to provide quite reliable forecasts. If some of the relevant conditional probabilities exhibit linearity in the sense that they are (approximately) linear functions in the variables upon which they are given, then a hybrid method might be advantageous. Assume, for example, that $p(X_v \,|\, X_{\mathrm{pa}(v)})p(X_{\mathrm{pa}(v)}) \approx p(X_v \,|\, X_{\mathrm{pa}(v)})p(X_u) \cdots p(X_w)$, where $\mathrm{pa}(v) = \{u, \ldots, w\}$. Then an estimate of $p(X_v)$ may be calculated as

$$\hat{p}(X_v) = \sum_{u,\ldots,w} p(X_v \,|\, X_{\mathrm{pa}(v)}) \hat{p}(X_u) \cdots \hat{p}(X_w),$$

where $\hat{p}(X_u), \ldots, \hat{p}(X_w)$ have been obtained via Monte-Carlo simulation or in a way similar to $\hat{p}(X_v)$.

7 Related work

In classical time-series analysis, for example, Box & Jenkins (1976) or West & Harrison (1989), the emphasis is on dynamic modelling, i.e., model assessment through estimation of model parameters given a time series of observations of some stochastic process. The selected model is then used for making predictions about the future behaviour of the time series. Although the classical time-series analysis techniques have been quite successful, their ability to cope with such important issues as complex independence structures and non-linear relationships have appeared to be rather modest. By formulating the analysis in terms of DBNs both of these limitations vanish.

It should be stressed, however, that the current version of the dHugin system lacks the ability to make model assessment; it merely addresses the computational issues of dynamic time-sliced reasoning. That is, changes in a time series caused by unmodelled exogenous events must be reflected through the specification of the time-slice networks (which may be modified dynamically by some external agent) before entered into the dHugin system.

Attempts to integrate methods of classical time-series analysis with Bayesian network representation and inference techniques have been presented by Kenley (1986) and Dagum et al. [Dagum et al. (1992), Dagum & Galper (1993a), Dagum & Galper (1993b)]. The element from classical time-series analysis represented in the method of Dagum et al. concerns estimation of a single parameter allowing the dynamic model to adapt to unexpected changes in the time series. The parameter controls the relative strengths of contemporaneous and non-contemporaneous influences (i.e., the relative importance of past and present observations for prediction of present and future variables). Thus, sudden changes in the time series caused by unmodelled exogenous events are handled through weakening of the non-contemporaneous influences (i.e., reducing the influence of past observations). The forecast method employed in their approach is the linear approximation method described in Section 6.3 [Dagum & Galper (1993b)].

Berzuini et al. (1989) embed semi-Markov models in a Bayesian network and use Monte-Carlo sampling schemes for inference. Dean & Kanazawa (1989) employs time-sliced Bayesian networks for making judgements about the persistence of propositions with time. Their forecast model for reasoning about persistence is based on survivor functions with exponential decay. Kanazawa (1991) combines a formal declarative language based on temporal logics with Bayesian networks for inference. However, none of these activities address the problem of developing a generic framework for dynamic time-sliced reasoning in Bayesian networks.

8 Conclusion

We have presented a computational system for reasoning in dynamic time-sliced Bayesian networks, featuring description of non-linear, multivariate dynamic systems with complex conditional independence structures and providing mechanisms for efficient forward propagation (incorporation of late observations) and backward propagation (backward smoothing).

As opposed to a static network representing a fixed number of time slices (i.e., capable of reasoning only about a finite series of observations of a dynamic system) the proposed system can handle infinite series of observations. Further, in applying static networks representing a fixed number of time slices as models of dynamic systems, there is typically a desire to include as many time slices as possible in the model. Thus, inference easily becomes time consuming and inflexible (i.e., propagation involves all time slices in the model even if updated distributions are wanted only for a limited number of time slices). The proposed system, on the other hand, provides a high degree of flexibility in the reasoning process, since the width of the window of time slices can be changed dynamically as well as the number of 'backward smoothing slices' and the number of 'forecast slices'. In addition, the system provides selective inference in the sense that inference can be performed (i) in the window, (ii) as forward propagation, (iii) as backward smoothing, or (iv) as forecasting.

Although we have presented a system for reasoning in time-sliced networks, some important issues have not been addressed. The most important seems to be the lack of automatic model assessment (compare discussion in Section 7).

Among the applications of DBNs for solving real-world problems is a system for glucose prediction and insulin dose adjustment by Andreassen et al. (1991), an approach to building planning and control systems by Dean et al. (1990), a model for sensor validation by Nicholson & Brady (1992), and a preliminary system for forecasting the development of fungus and estimating/forecasting the gross yield from a field of wheat (Fig. 2).

An implementation, dHugin, of the computational system presented has been built on top of the Hugin shell [Andersen et al. (1989)]. The implementation is provided as an extension of the regular Hugin API (application program interface) [Kjærulff (1993b)]. In addition, based on the extended API library, a menu-driven user interface has been built [Kjærulff (1993c)]. This interface provides facilities for defining a DBN (through specification of conditional time-sliced models, number of forecast slices, etc.), moving the window forward and backward, changing the window width and the number of forecast and backward smoothing slices, updating beliefs in the window, performing backward and forward propagation, making forecasts via forward sampling, plus the usual facilities of inserting and retracting evidence and viewing beliefs of single variables. Access to the dHugin system can be provided for Hugin license holders only.

9 Acknowledgements

Jørgen E. Olesen and Finn V. Jensen played a central role in the development of the (preliminary) time-sliced wheat network of Fig. 2. Steffen L. Lauritzen and Finn V. Jensen provided valuable comments on an earlier draft of the paper. The research has been funded partly by the Danish Research Councils through the PIFT programme.

10 References

Andersen, S. K., K. G. Olesen, F. V. Jensen and F. Jensen, 1989, "HUGIN — A shell for building Bayesian belief universes for expert systems", *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1080–1085.

Andreassen, S. A., M. Woldbye, B. Falck and S. K. Andersen, 1987, "MUNIN – A causal probabilistic network for interpretation of electromyographic findings", *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, 366–372.

Andreassen, S., R. Hovorka, J. Benn, K. G. Olesen and E. R. Carson, 1991, "A model-based approach to insulin adjustment", in: M. Stefanelli, A. Hasman, M. Fieschi and J. Talmon eds., *Proceedings of the Third Conference on Artificial Intelligence in Medicine* (Springer-Verlag), 239–248.

Arnborg, S., D. G. Corneil and A. Proskurowski, 1987, "Complexity of finding embeddings in a k-tree", *SIAM Journal on Algebraic and Discrete Methods*, 8, 277–284.

Berzuini, C., R. Bellazzi and S. Quaglini, 1989, "Temporal reasoning with probabilities", *Proceedings of the Fifth Workshop on Uncertainty in Artificial Intelligence*, Association for Uncertainty in Artificial Intelligence, 14–21.

Box, G. E. P. and G. M. Jenkins, 1976, *Time Series Analysis: Forecasting and Control*, Holden-Day Series in Time Series Analysis and Digital Processing (Holden-Day, San Francisco, California).

Cooper, G. F., 1990, "The computational complexity of probabilistic inference using Bayesian belief networks", *Artificial Intelligence*, 42, 393–405.

Dagum, P. and A. Galper, 1993a, "Additive belief-network models", *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann Publishers, San Mateo, California), 91–98.

Dagum, P. and A. Galper, 1993b, "Forecasting sleep apnea with dynamic network models", *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann Publishers, San Mateo, California), 64–71.

Dagum, P., A. Galper and E. Horvitz, 1992, "Dynamic network models for forecasting", *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann Publishers, San Mateo, California), 41–48.

Dawid, A. P., 1992, Applications of a general propagation algorithm for probabilistic expert systems, *Statistics and Computing*, 2, 25–36.

Dean, T., K. Basye and M. Lejter, 1990, "Planning and active perception", *Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control*, 271–276.

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

Heckerman, D., E. Horvitz and B. Nathwani, 1992, "Toward normative expert systems: Part I. The Pathfinder project", *Methods of Information in Medicine*, 31, 90–105.

Henrion, M., 1988, "Propagating uncertainty in Bayesian networks by probabilistic logic sampling", in: J. F. Lemmer and L. M. Kanal eds., *Uncertainty in Artificial Intelligence* (Elsevier Science Publishers B. V. (North-Holland), Amsterdam), 149–163.

Jensen, F. V., 1988, "Junction trees and decomposable hypergraphs", *Research report*, Judex Datasystemer A/S, Aalborg, Denmark.

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

Kanazawa, K., 1991, "A logic and time nets for probabilistic inference", *Proceedings of the Tenth National Conference on Artificial Intelligence*, American Association for Artificial Intelligence, 360–365.

Kenley, C. R., 1986, *Influence Diagram Models with Continuous Variables*, PhD thesis, Department of Engineering-Economic Systems, Stanford University, California.

Kjærulff, U., 1992, "Optimal decomposition of probabilistic networks by simulated annealing", *Statistics and Computing*, 2, 7–17.

Kjærulff, U., 1993a, "A computational scheme for dynamic Bayesian networks", *Research Report R-93-2018*, Department of Mathematics and Computer Science, Aalborg University, Denmark.

Kjærulff, U., 1993b, "dHUGIN API reference manual", *Technical Report IR-93-2004*, Department of Mathematics and Computer Science, Aalborg University, Denmark.

Kjærulff, U., 1993c, "User's guide to `dhugin`", *Technical Report IR-93-2005*, Department of Mathematics and Computer Science, Aalborg University, Denmark.

Lauritzen, S. L., A. P. Dawid, B. N. Larsen and H.-G. Leimer, 1990, "Independence properties of directed Markov fields", *Networks*, 20(5), 491–505. Special Issue on Influence Diagrams.

Lauritzen, S. L., T. P. Speed and K. Vijayan, 1984, "Decomposable graphs and hypergraphs", *Journal of The Australian Mathematical Society, A*, 36, 12–29.

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

Miller, R. A., H. E. Pople and J. Myers, 1982, "Internist-1, an experimental computer-based diagnostic consultant for general internal medicine", *New England Journal of Medicine*, 307, 468–476.

Nicholson, A. E. and J. M. Brady, 1992, "Sensor validation using dynamic belief networks", *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence* (Morgan Kaufmann Publishers, San Mateo, California), 207–214.

Pearl, J., 1988, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Series in Representation and Reasoning (Morgan Kaufmann Publishers, San Mateo, California).

Rose, D. J., 1973, "A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations", in: R. C. Read ed., *Graph Theory and Computing* (Academic Press, New York), 183–217.

Rose, D. J., R. E. Tarjan and G. S. Lueker, 1976, "Algorithmic aspects of vertex elimination on graphs", *SIAM Journal on Computing*, 5, 266–283.

Shafer, G. and P. P. Shenoy, 1990, "Probability propagation", *Annals of Mathematics and Artificial Intelligence*, 2, 327–352.

Spiegelhalter, D. J., 1986, "Probabilistic reasoning in predictive expert systems", in: J. F. Lemmer and L. M. Kanal eds., *Uncertainty in Artificial Intelligence* (Elsevier Science Publishers B. V. (North-Holland), Amsterdam).

Spiegelhalter, D. J., A. P. Dawid, S. L. Lauritzen and R. G. Cowell, 1993, "Bayesian analysis in expert systems" (with discussion), *Statistical Science*, 8, 219–247 and 204–283.

Swhe, M., B. Middleton, D. Heckerman, M. Henrion, E. Horvitz and H. Lehmann, 1991, "Probabilistic diagnosis using a reformulation of the INTERNIST-1/QMR knowledge base I: The probabilistic model and inference algorithms", *Methods of Information in Medicine*, 30, 241–255.

West, M. and J. Harrison, 1989, *Bayesian Forecasting and Dynamic Models*, Series in Statistics (Springer-Verlag, New York).

11 Biography

Uffe Kjærulff is an assistant professor of computer science at Aalborg University. He received his M.Sc. degree in computer engineering from Aalborg University in 1985 and his Ph.D. from the same university in 1993. From 1985 to 1989 he was a research assistant on the MUNIN project which aimed at developing an EMG (electromyography) expert assistant for diagnosing disorders in the human peripheral nervous system. He is a co-author of a number of papers on the MUNIN system. His research has been in the area of improving the efficiency of inference in Bayesian (probabilistic) networks. Recently, his work has concentrated on the use of Monte-Carlo simulation in Bayesian networks.