An Extension of the Differential Approach for Bayesian Network Inference to Dynamic Bayesian Networks

Boris Brandherm
Department of Computer Science, Saarland University
P. O. Box 15 11 50, D-66041 Saarbrücken, Germany
brandherm@cs.uni-sb.de

Abstract

We extend the differential approach to inference in Bayesian networks (BNs) (Darwiche, 2000) to handle specific problems that arise in the context of dynamic Bayesian networks (DBNs). We first summarize Darwiche's approach for BNs, which involves the representation of a BN in terms of a multivariate polynomial. We then show how procedures for the computation of corresponding polynomials for DBNs can be derived. These procedures permit not only an exact roll-up of old time slices but also a constant-space evaluation of DBNs. The method is applicable to both forward and backward propagation, and it does not presuppose that each time slice of the DBN has the same structure. It is compatible with approximative methods for roll-up and evaluation of DBNs.

Introduction

Dynamic Bayesian networks (DBNs) are an extension of Bayesian networks (BNs). With a DBN, it is possible to model dynamic processes: Each time the DBN receives new evidence, a new time slice is added to the existing DBN. Each time slice must satisfy the Markov assumption: The future is conditionally independent of the past given the current state. In principle, DBNs can be evaluated with the same inference procedures as normal BNs; but their dynamic nature places heavy demands on computation time and memory. Therefore, it is necessary to apply roll-up procedures that cut off old time slices without eliminating their influence on the newer time slices. For this condition to be fulfilled, a probability table representing the *belief state* has to be maintained: a probability distribution over the state of the system at a given time. This belief state includes at least all of the nodes that are parents of nodes in the next time slice. The representation of this belief state as a probability table raises complexity problems: This table cannot in general be given a factorized representation without loss of information (see, e.g., Boyen & Koller, 1998). Roll-up procedures may be exact (see, e.g., Kjærulff, 1995) or approximative (see, e.g., Boyen & Koller, 1999).

With his differential approach to inference in Bayesian networks, Darwiche (2000) presents an algorithm that compiles a BN into a multivariate polynomial that can be processed efficiently: After the polynomial and its partial derivatives have been computed, a large class of probabilistic computations, such as classical inference and sensitivity analysis, can be computed with constant time.

In this paper, we extend the differential approach so that DBNs can be handled relatively efficiently, like (static) BNs. In particular, once an approximate factorization of the belief state has been computed, the effects of this approximation on the values of particular nodes can be analyzed with the differential approach. For example, with sensitivity analysis, we can estimate the influence of evidence on newer time slices, so that, if necessary, less important network structures can be cut off.

In Section , we will summarize the approach of Darwiche (2000) for BNs, looking at the canonical and factorial representation of the multivariate polynomials that represent a BN. We will indicate which queries to the BN can be answered with the polynomial. In Section 2 we will show how the approach can be extended to DBNs, developing a procedure within the differential approach that allows us perform to roll-ups and perform inference in DBNs with constant space requirements.

Polynomial Representation of Bayesian Networks

A BN $\mathcal{N}$ can be represented graphically as a directed acyclic graph (see Figure 1). The nodes of the graph represent the variables[1] $X_1, \ldots, X_n$. The edges joining the nodes represent the dependencies among them. For each node X_i, there is a conditional probability table (CPT) $\boldsymbol{\theta}_{(X_i|\mathbf{pa}(X_i))}$ that quantifies these dependencies. (The parent nodes of X_i are denoted by $\mathbf{pa}(X_i)$.) A number of different procedures exist for evaluating BNs (see, e.g., Pearl, 1988; Jensen, 2001).

Canonical Representation of the Polynomial

Any BN that contains only discrete variables can be represented by a polynomial. In Figure 1, for example, let us look first only at the binary nodes A and B and their associated CPTs $\boldsymbol{\theta}_{(A)}$ and $\boldsymbol{\theta}_{(B|A)}$. The two possible values of A are denoted by a_1 and a_2, while those of B are denoted by b_1 and b_2. Suppose, for example, that we have evidence $\mathbf{e} = a_1$ and that we are interested in $\Pr(\mathbf{e})$. By multiplying the two

[1]In this paper, we use the terms "node" and "variable" interchangeably.

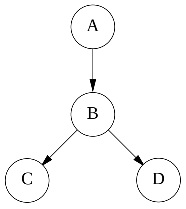

Figure 1: Example of a Bayesian network with four binary nodes [$\Pr(A, B, C, D) = \Pr(A) * \Pr(B \mid A) * \Pr(C \mid B) * \Pr(D \mid B)$]

CPTs, we obtain an overall probability table $\boldsymbol{\theta}_{(A,B)}$ that represents the joint probability distribution of the nodes A and B:

$$\boldsymbol{\theta}_{(A,B)} = \boldsymbol{\theta}_{(A)}\ \boldsymbol{\theta}_{(B|A)}$$
$$= \begin{bmatrix} \theta_{a_1} \\ \theta_{a_2} \end{bmatrix} \begin{bmatrix} \theta_{b_1|a_1} & \theta_{b_1|a_2} \\ \theta_{b_2|a_1} & \theta_{b_2|a_2} \end{bmatrix} = \begin{bmatrix} \theta_{a_1}\theta_{b_1|a_1} & \theta_{a_2}\theta_{b_1|a_2} \\ \theta_{a_1}\theta_{b_2|a_1} & \theta_{a_2}\theta_{b_2|a_2} \end{bmatrix} .$$

Now we need only to add up the table entries (probabilities) that are consistent with a_1, and we obtain the desired result:

$$\Pr(\mathbf{e}) = \theta_{a_1}\theta_{b_1|a_1} + \theta_{a_1}\theta_{b_2|a_1} .$$

For each variable in the BN, it is possible that evidence has been obtained. Such evidence can be represented by an *evidence vector* λ_{X_i} for the node in question. The vector contains a 1 for the state which is realized and a 0 for each other state of the variable. (When no evidence concerning a given variable is available, each element of the evidence vector is 1.)

We can use the evidence vectors to parametrize the probability table, so that it takes into account the available evidence, as follows:

$$\boldsymbol{\theta}_{(A,B)}\lambda_A\lambda_B = \boldsymbol{\theta}_{(A)}\lambda_A\ \boldsymbol{\theta}_{(B|A)}\lambda_B$$
$$= \begin{bmatrix} \theta_{a_1}\theta_{b_1|a_1}\lambda_{a_1}\lambda_{b_1} & \theta_{a_2}\theta_{b_1|a_2}\lambda_{a_2}\lambda_{b_1} \\ \theta_{a_1}\theta_{b_2|a_1}\lambda_{a_1}\lambda_{b_2} & \theta_{a_2}\theta_{b_2|a_2}\lambda_{a_2}\lambda_{b_2} \end{bmatrix} .$$

By adding up all of the table entries, we obtain a multivariate polynomial:

$$\mathcal{F}(\lambda_A, \lambda_B, \boldsymbol{\theta}_{(A)}, \boldsymbol{\theta}_{(B|A)}) =$$
$$\theta_{a_1}\theta_{b_1|a_1}\lambda_{a_1}\lambda_{b_1} + \theta_{a_2}\theta_{b_1|a_2}\lambda_{a_2}\lambda_{b_1} +$$
$$\theta_{a_1}\theta_{b_2|a_1}\lambda_{a_1}\lambda_{b_2} + \theta_{a_2}\theta_{b_2|a_2}\lambda_{a_2}\lambda_{b_2} .$$

Now we can determine through the evidence vectors which table entries are to be added up. The λ_{x_i} that are consistent with the evidence $\mathbf{e}$ are set to 1, while the other λ_{x_i} are set to 0. In our example, therefore, $\lambda_{a_2} = 0$ and $\lambda_{a_1} = \lambda_{b_1} = \lambda_{b_2} = 1$.

With this polynomial, for example, we can compute the belief for particular nodes in the network, and we can perform various sensitivity analyses. Much more detail can be found in the articles by Darwiche.

In general, the multivariate polynomial is computed as follows: First we multiply all of the CPTs of the nodes of the BN together with their evidence vectors λ_{X_i} to obtain a table $\mathbf{T}$ that contains the probabilities of the individual hypothesis combinations, that is:

$$\mathbf{T} = \Pi_{i=1}^{n}\ \boldsymbol{\theta}_{(X_i|\mathbf{pa}(X_i))}\lambda_{X_i} .$$

Adding up all of the table entries in $\mathbf{T}$, we obtain the multivariate polynomial in canonical from (see also Darwiche, 2000).

$$\mathcal{F}(\lambda_{X_i}, \boldsymbol{\theta}_{(X_i|\mathbf{pa}(X_i))}) = \Sigma_{\mathbf{X}}\Pi_{i=1}^{n}\ \boldsymbol{\theta}_{(X_i|\mathbf{pa}(X_i))}\lambda_{X_i} .$$

The size of a multivariate polynomial in canonical form is always exponential in the number of the variables. A way of avoiding the resulting complexity is offered by the factored representation of the polynomial, which is discussed in the following section.

Factored Representation of the Polynomial

In finding a factorization of the multivariate polynomial, we want to take into account the structure of the Bayesian network; that is, we want to exploit the factored representation of the joint probability distribution in order to reduce the complexity of the multivariate polynomial. We will not simply compute the product of all of the CPTs with their evidence vectors. Instead, we will first find an ordering of the nodes according to which the CPTs and their corresponding evidence vectors λ_{X_i} are to be multiplied. Early in this process, we will marginalize as many nodes as possible out of this product, so as to obtain the multivariate polynomial in factored instead of canonical form.

As an illustration, let us take once again the example of the nodes A and B. Suppose that the order of processing of the nodes is "First B, then A". This order is also called the *elimination order* $\pi = \langle B, A \rangle$. If we marginalize out the nodes as soon as possible, we obtain:

$$\mathcal{F}(\lambda_A, \lambda_B, \boldsymbol{\theta}_{(A)}, \boldsymbol{\theta}_{(B|A)}) =$$
$$\mathcal{F}_{\langle B,A \rangle}(\lambda_A, \lambda_B, \boldsymbol{\theta}_{(A)}, \boldsymbol{\theta}_{(B|A)}) =$$
$$\Sigma_A\ \boldsymbol{\theta}_{(A)}\lambda_A\ (\Sigma_B\ \boldsymbol{\theta}_{(B|A)}\lambda_B) =$$
$$\theta_{a_1}\lambda_{a_1}\ (\theta_{b_1|a_1}\lambda_{b_1} + \theta_{b_2|a_1}\lambda_{b_2}) +$$
$$\theta_{a_2}\lambda_{a_2}\ (\theta_{b_1|a_2}\lambda_{b_1} + \theta_{b_2|a_2}\lambda_{b_2}) .$$

When we compare the polynomial in factored form with the polynomial in canonical form, we notice that in the factored form there are some multiplications less that need to be computed. In connection with the notation of the elimination order, it should be noted that the following holds:

$$\Pi_{\langle B,A \rangle}\ \boldsymbol{\theta}_{(X|\mathbf{pa}(X))}\lambda_X = \boldsymbol{\theta}_{(A)}\lambda_A\ \boldsymbol{\theta}_{(B|A)}\lambda_B$$

and

$$\Sigma_{\langle B,A \rangle}\ \boldsymbol{\theta}_{(A)}\lambda_A\ \boldsymbol{\theta}_{(B|A)}\lambda_B =$$
$$\Sigma_{\langle A \rangle}\Sigma_{\langle B \rangle}\ \boldsymbol{\theta}_{(A)}\lambda_A\ \boldsymbol{\theta}_{(B|A)}\lambda_B =$$
$$\Sigma_A\ \boldsymbol{\theta}_{(A)}\lambda_A\ (\Sigma_B\ \boldsymbol{\theta}_{(B|A)}\lambda_B).$$

We will use the following abbreviated notation for the multivariate polynomial in factored form:

$$\mathcal{F}_{\mathrm{elim}(\{X \in \mathcal{N}\})}(\lambda_X, \boldsymbol{\theta}_{(X|\mathbf{pa}(X))}) =$$
$$\Sigma_{\mathrm{elim}(\{X \in \mathcal{N}\})}\ \Pi_{\mathrm{elim}(\{X \in \mathcal{N}\})}\ (\boldsymbol{\theta}_{(X|\mathbf{pa}(X))}\lambda_X) =$$
$$\Sigma_{\langle X_1,\ldots,X_n \rangle}\ \Pi_{\langle X_1,\ldots,X_n \rangle}\ (\boldsymbol{\theta}_{(X|\mathbf{pa}(X))}\lambda_X) .$$

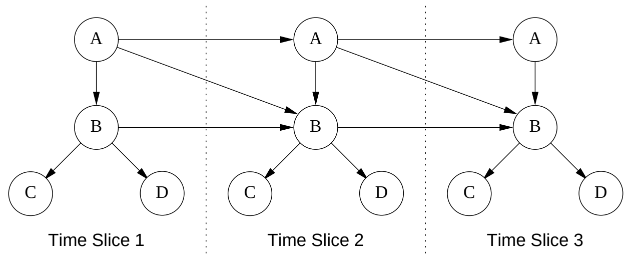

Figure 2: Example dynamic Bayesian network discussed in the text [$\Pr(A_1, B_1, \ldots) = \Pr(A_1) * \Pr(B_1 \mid A_1) * \ldots$]

Here, $\text{elim}(\{X \in \mathcal{N}\})$ is a function that computes the elimination order of the variables.[2]

Computation of the Polynomial in DBNs

Now we will see how these ideas of Darwiche (2000) can be extended and applied to DBNs. Let us look at the DBN in Figure 2, which at this point comprises three time slices. This DBN includes two building blocks (so-called *time slice schemas*), shown in Figure 3. The first schema, used for the first time slice, is simply a BN. The second schema, which is used for each of the remaining time slices, is a BN plus a specification of the parent variables that are taken from the preceding time slice. Suppose that we have evidence $\mathbf{e} = \mathbf{e}_1, \ldots, \mathbf{e}_3$ for Time Slices 1 through 3 and that we want to know the *belief vector* for a node in Time Slice 2—i.e., the vector that expresses how likely each possible value of the variable is, given the currently available evidence.

We now have to distinguish between (a) forward propagation, which brings forward the impact of the evidence from Time Slice 1 to Time Slice 2; and (b) backward propagation, which brings the impact of evidence from Time Slice 3 back to Time Slice 2.

In the following we will define procedures that will allow us to determine the beliefs for nodes in an arbitrary time slice t (with $1 \leq t \leq L$); to cut off old time slices; and to add new time slices. These computations can be performed in constant space (depending on the structure of the time slice schemas). We perform a roll-up by assigning values to the variables of the polynomial and then simplifying the polynomial by evaluating it. We add new time slices efficiently by recycling polynomials and computations from the preceding time slices.

Typically, we will not use a single arbitrary elimination order for the whole DBN; instead we will use an elimination order that is restricted to one time slice. In this way, we will obtain for each time slice schema a general procedure for the partial polynomial that corresponds to this time slice schema.

Forward Propagation

First, we will look at the case of forward propagation. To simplify exposition, we assume that we have only one time slice schema available for the instantiation of the initial time slice, and also only one time slice schema for the instantiation of the following time slices, as in our example DBN. Later, we will sketch how it is possible to generalize the procedure to deal with an arbitrary number of time slice schemas for the instantiation of the initial and the succeeding time slices.

[2]The question of which such function is to be used does not concern us here; cf Kjærulff (1995).

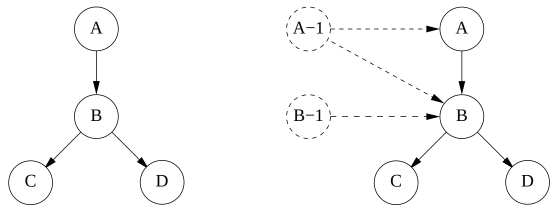

Figure 3: Time Slice Schemas 1 and 2 for the dynamic Bayesian network shown in Figure 2

For the first time slice, we will determine a procedure that permits a simple and efficient extension of the old polynomial when a new time slice is added to the DBN. Let us look at Time Slices 1 and 2 in Figure 2. We can see that in Time Slice 1 the parent nodes of nodes in Time Slice 2 cannot be marginalized out until Time Slice 2 has been added. The result of the procedure for the first time slice is a table over the nodes of the current time slice that could not be marginalized out. Because only one time slice schema can be instantiated for the following time slices, we know which nodes in the current time slice will become parent nodes of nodes in the following time slices, and we can determine the nodes that belong to the belief state. (The set of these nodes is denoted by $\mathbf{bs}(.)$.) In our example, these are the nodes A_1 and B_1, so $\mathbf{bs}(1) = \{A_1, B_1\}$. The indices in the nodes denote the time slice to which they belong.

In our example DBN, we obtain as a procedure for forward propagation for the first time slice:

$$\begin{aligned}\text{fwd}_{\text{init}} = & \, \boldsymbol{\theta}_{(A_1)}\lambda_{A_1} \, \boldsymbol{\theta}_{(B_1|A_1)}\lambda_{B_1} \\ & (\Sigma_{C_1} \, \boldsymbol{\theta}_{(C_1|B_1)}\lambda_{C_1}) \, (\Sigma_{D_1} \, \boldsymbol{\theta}_{(D_1|B_1)}\lambda_{D_1}) \, .\end{aligned}$$

As was already mentioned, the result is a table over the Cartesian product of the hypotheses of the nodes A_1 and B_1, that is $\text{fwd}_{\text{init}} = \mathbf{T}_{(A_1 B_1)}$. The table entries of $\mathbf{T}_{(A_1 B_1)}$ itself are polynomials. We obtain the polynomial for the first time slice by summing over all of these table entries, that is over all of the nodes that have been left out so far. With this polynomial, it is possible, for example, to compute the belief value of a node. In $\mathbf{T}_{(A_1 B_1)}$, on the other hand, all of the information is contained that must be passed to a later time slice so as to permit exact inference. As yet no evidence has been instantiated in the polynomial or in $\mathbf{T}_{(A_1 B_1)}$. To simplify the polynomial or $\mathbf{T}_{(A_1 B_1)}$, all of the λ_{x_i} of the noninteresting nodes can be set to the given evidence or to 1 if no evidence is given.

Now let us look at Time Slice 2 with its predecessor Time Slice 1 and its successor Time Slice 3 in Figure 2. We want to determine a procedure which takes into account the effects of the earlier time slice on the following time slices and which allows a simple and efficient extension of the old polynomial when a new time slice is added.

As was the case with the procedure for the first time slice, the nodes of the belief state cannot be marginalized out until the following time slice has been added. In our example, the nodes in question are A_2 and B_2.

We obtain the following as a procedure for forward propagation in the second time slice in our example in Figure 2:

$$\begin{aligned}&\mathrm{fwd}_2(\mathbf{T}_{(A_1B_1)}) = \\ &\quad (\Sigma_{C_2}\ \boldsymbol{\theta}_{(C_2|B_2)}\lambda_{C_2})\ (\Sigma_{D_2}\ \boldsymbol{\theta}_{(D_2|B_2)}\lambda_{D_2}) \\ &\quad (\Sigma_{A_1}\ \boldsymbol{\theta}_{(A_2|A_1)}\lambda_{A_2}\ (\Sigma_{B_1}\ \boldsymbol{\theta}_{(B_2|A_2,A_1,B_1)}\lambda_{B_2} \\ &\qquad \mathbf{T}_{(A_1B_1)})) = \mathbf{T}_{(A_2B_2)}\ .\end{aligned}$$

It is easy to see that the procedures for the following time slices are all identical and that the procedures for the time slices i (with $i \geq 1$) can be generalized to:

$$\begin{aligned}&\mathrm{fwd}_i(\mathbf{T}_{(A_{i-1}B_{i-1})}) = \\ &\quad (\Sigma_{C_i}\ \boldsymbol{\theta}_{(C_i|B_i)}\lambda_{C_i})\ (\Sigma_{D_i}\ \boldsymbol{\theta}_{(D_i|B_i)}\lambda_{D_i}) \\ &\quad (\Sigma_{A_{i-1}}\ \boldsymbol{\theta}_{(A_i|A_{i-1})}\lambda_{A_i}\ (\Sigma_{B_{i-1}}\ \boldsymbol{\theta}_{(B_i|A_i,A_{i-1},B_{i-1})}\lambda_{B_i} \\ &\qquad \mathbf{T}_{(A_{i-1}B_{i-1})})) = \mathbf{T}_{(A_iB_i)}\ .\end{aligned}$$

In the case where $i = 1$, the variables with the index 0 disappear, and we have:

$$\mathrm{fwd}_1(\mathbf{T}_{(A_0B_0)}) = \mathrm{fwd}_1(1).$$

The polynomial for Time Slices 1 through L is as follows:

$$\begin{aligned}&\mathcal{F}_{\mathrm{elim}(\{X\in\mathcal{N}\})}(\lambda_{X_i}, \boldsymbol{\theta}_{(X_i|\mathbf{pa}(X_i))}) = \\ &\Sigma_{\mathrm{elim}(\{A_L,B_L\})}(\mathrm{fwd}_L(\ldots\mathrm{fwd}_2(\mathrm{fwd}_1(1))\ldots)) = \\ &\Sigma_{\mathrm{elim}(\{A_L,B_L\})}(\mathrm{fwd}_L \circ \ldots \circ \mathrm{fwd}_2 \circ \mathrm{fwd}_1(1))\ .\end{aligned}$$

Before we show how the polynomial can be evaluated with constant space requirements, we want to capture the results obtained so far in a general notation. As a general procedure for forward propagation from the first time slice, we obtain:

$$\begin{aligned}\mathrm{fwd}_{\text{init}} = {}& \Sigma_{\mathrm{elim}(\{X|X\in\mathrm{TS}(1)\setminus\mathbf{bs}(1)\})} \\ & \Pi_{X\in\mathrm{TS}(1)}\ \boldsymbol{\theta}_{(X|\mathbf{pa}(X))}\lambda_X = \\ & \Sigma_{\mathrm{elim}(\{X|X\in\mathrm{TS}(1)\setminus\mathbf{bs}(1)\})} \\ & \Pi_{X\in\mathrm{TS}(1)}\ \boldsymbol{\theta}_{(X|\mathbf{pa}(X))}\lambda_X\ 1 = \mathrm{fwd}_1(1)\ .\end{aligned}$$

$\mathrm{TS}(1)$ denotes the nodes of the first time slice, and $\mathbf{bs}(1)$ denotes the set of nodes that belong to the belief state of the first time slice. To simplify exposition (especially for the algorithm in the next column) we have introduced here the notation $\mathrm{fwd}_1(1)$ for $\mathrm{fwd}_{\text{init}}$.

The polynomial for the first time slice is therefore as follows:

$$\begin{aligned}&\mathcal{F}_{\mathrm{elim}(\{X\in\mathcal{N}\})}(\lambda_{X_i}, \boldsymbol{\theta}_{(X_i|\mathbf{pa}(X_i))}) = \\ &\Sigma_{\mathrm{elim}(\mathbf{bs}(1))}(\mathrm{fwd}_{\text{init}}) = \\ &\Sigma_{\mathrm{elim}(\mathbf{bs}(1))}(\mathrm{fwd}_1(1)) = \\ &\Sigma_{\mathrm{elim}(\mathbf{bs}(1))}(\Sigma_{\mathrm{elim}(\{X|X\in\mathrm{TS}(1)\setminus\mathbf{bs}(1)\})} \\ &\qquad \Pi_{X\in\mathrm{TS}(1)}\ \boldsymbol{\theta}_{(X|\mathbf{pa}(X))}\lambda_X\ 1)\ .\end{aligned}$$

Now let us look at the case in which a time slice is to be added to the existing DBN. The situation is different from the one for the first time slice: To the general procedure for the i-th time slice is now passed on the table over all nodes that were not marginalized out by the procedure that was just applied:

$$\begin{aligned}\mathrm{fwd}_i(\mathbf{T}) = {}& \Sigma_{\mathrm{elim}(\{X|X\in\mathrm{TS}(i)\setminus\mathbf{bs}(i)\})} \\ & \Sigma_{\mathrm{elim}(\{X|X\in\mathbf{bs}(i-1)\})} \\ & \Pi_{X\in\mathrm{TS}(i)}\ \boldsymbol{\theta}_{(X|\mathbf{pa}(X))}\lambda_X\ \mathbf{T}\ .\end{aligned}$$

$\mathbf{T}$ is a table over the set of nodes $\{X \mid X \in \mathbf{bs}(i-1)\}$, which in general was computed by the procedure of the preceding time slice. The result of the function fwd_i is a table over the set of nodes $\{X \mid X \in \mathbf{bs}(i)\}$.

The polynomial for Time Slices 1 through L is as follows:

$$\begin{aligned}&\mathcal{F}_{\mathrm{elim}(\{X\in\mathcal{N}\})}(\lambda_{X_i}, \boldsymbol{\theta}_{(X_i|\mathbf{pa}(X_i))}) = \\ &\Sigma_{\mathrm{elim}(\mathbf{bs}(L))}(\mathrm{fwd}_L(\ldots\mathrm{fwd}_2(\mathrm{fwd}_1(1))\ldots)) = \\ &\Sigma_{\mathrm{elim}(\mathbf{bs}(L))}(\mathrm{fwd}_L \circ \ldots \circ \mathrm{fwd}_2 \circ \mathrm{fwd}_1(1))\ .\end{aligned}$$

The evaluation of the polynomial with constant space requirements for the time slice L is obtained as follows:

1. $\mathbf{T} = 1$;
2. For $i = 1$ to L:
   $\mathbf{T}_{\text{new}} = \mathrm{fwd}_i(\mathbf{T})$; $\quad \mathbf{T} = \mathbf{T}_{\text{new}}$;
3. $\mathcal{F}(\mathbf{e}_1, \ldots, \mathbf{e}_L) = \Sigma_{\mathrm{elim}(\mathbf{bs}(L))}(\mathbf{T})$;

In Step 1, for the initial time slice, the value 1 is passed on to the table $\mathbf{T}$. In Step 2, a loop is traversed with the index running from 1 to L. In the first iteration, a new table $\mathbf{T}_{\text{new}}$ is computed by the initial time slice, that is $\mathbf{T}_{\text{new}} = \mathrm{fwd}_1(1)$. In the i-th iteration, $\mathbf{T}_{\text{new}} = \mathrm{fwd}_i(T)$ is computed, which takes into account the effects of the earlier time slice. In the computation within the loop, already existing variables are reused efficiently. In particular, for the determination of $\mathrm{fwd}_i(\mathbf{T})$, the corresponding general procedures are applied. Consequently, in a subsequent pass through the loop, no further memory allocation is performed if all evidence vectors of the current time slice have been given numerical values. Finally, in step 3 the polynomial is computed.

The polynomial $\mathcal{F}(\mathbf{e}_1, \ldots, \mathbf{e}_L)$ can be used for the computation of the belief values of the nodes in time slice t, as well as for other computations, such as sensitivity analyses, as was mentioned briefly in Section 1 and as is discussed in more detail by Darwiche (2000).

The procedure can also be generalized to DBNs in which more than one time slice schema is available for the instantiation of each time slice. A directed graph can be used to model the dependencies among the time slice schemas that determine in which orders they can be instantiated, so that the number of procedures can be minimized. In Figure 4 the solid lines indicate the directed graph that models the dependencies among the time slice schemas (see Figure 3) for the example DBN (see Figure 2). The whole directed graph (with dashed and solid lines) in Figure 4 comprises two time slice schemas (Time Slice Schema 1 and Time Slice Schema 3) that can be instantiated as the initial time slice of the DBN and two other time slice schemas (Time Slice Schema 2 and Time Slice Schema 4) that can be instantiated as following time slices of the DBN. After Time Slice Schema 1, either

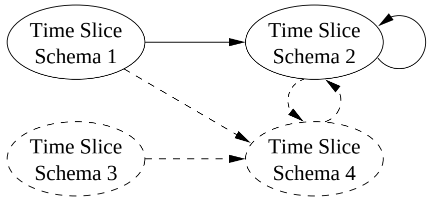

Figure 4: Directed graph that determines in which orders the time slice schemas can be instantiated

Time Slice Schema 2 or Time Slice Schema 4 can be instantiated as the following time slice. After Time Slice Schema 3, only Time Slice Schema 4 can be instantiated as the following time slice.

The procedures can be adapted so that we do not have to compute the table $\mathbf{T}$, which passes on all information without approximation. Instead, several smaller tables can be computed which are much less complex than the overall table $\mathbf{T}$ and whose product approximates the overall table. Boyen and Koller (1999) describe how the overall table can be split up into smaller tables with minimal loss of information.

Backward Propagation and Combined Forward and Backward Propagation

As was the case with forward propagation, general procedures for backward propagation can be determined that pass the effects of evidence from newer time slices on to older time slices.

If we are interested in the belief values of nodes in time slice t of a DBN from the Time Slices 1 through L (with $1 \leq t \leq L$), we can combine the procedures for forward and backward propagation: We multiply the procedures for the polynomial for forward propagation from Time Slice 1 through t with the procedures for the polynomial for backward propagation from Time Slices L through $t+1$. Then we marginalize out the nodes $\mathbf{bs}(t)$.

The evaluation of the polynomial with constant space requirements for the time slice t is obtained as follows:

```
1. $\mathbf{F} = 1$;
2. For $i = 1$ to $t$:
   $\mathbf{F}_{\text{new}} = \text{fwd}_i(\mathbf{F})$;   $\mathbf{F} = \mathbf{F}_{\text{new}}$;
3. $\mathbf{B} = 1$;
4. For $i = L$ downto $t + 1$:
   $\mathbf{B}_{\text{new}} = \text{bwd}_i(\mathbf{B})$;   $\mathbf{B} = \mathbf{B}_{\text{new}}$;
5. $\mathcal{F}(\mathbf{e}_1, \ldots, \mathbf{e}_t, \ldots, \mathbf{e}_L) = \Sigma_{\text{elim}(\mathbf{bs}(t))}(\mathbf{F} * \mathbf{B})$;
```

Here the same remarks hold as for the preceding algorithm. We now call the table $\mathbf{F}$ in the case of forward propagation und $\mathbf{B}$ in the case of backward propagation.

Summary

We have shown how the differential approach to the evaluation of Bayesian networks (see Darwiche, 2000) can be extended to dynamic Bayesian networks. We have specified the procedures that can be used to determine the relevant polynomials for arbitrarily large DBNs. Computations for partial polynomials can be reused.

Through the use of these formulas, we can perform forward and backward propagation (including the combination of the two). We can also roll up older time slices and other superfluous network structures while ensuring constant space requirements in the evaluation of the polynomials.

The other advantages of the differential approach are now also available for DBNs, for example, efficient methods for sensitivity analysis (see Darwiche, 2000, 1999).

The procedures involved can be translated into efficient machine code for the computation of belief values in a procedure analogous to the one proposed by Takikawa, D'Ambrosio, and Wright (2002).

Acknowledgements

This research is being supported by the German Science Foundation (DFG) in its Collaborative Research Center on Resource-Adaptive Cognitive Processes[3], SFB 378, Project B2, READY[4].

References

Boyen, X., & Koller, D. (1998). Tractable inference for complex stochastic processes. In G. F. Cooper & S. Moral (Eds.), *Uncertainty in Artificial Intelligence: Proceedings of the Fourteenth Conference* (pp. 33–42). San Francisco, CA: Morgan Kaufmann.

Boyen, X., & Koller, D. (1999). Exploiting the architecture of dynamic systems. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence* (pp. 313–320). Orlando, FL.

Darwiche, A. (1999). *A differential approach to inference in Bayesian networks* (Tech. Rep. Nos. D–108). Computer Science Department, UCLA, Los Angeles, Ca 90095.

Darwiche, A. (2000). A differential approach to inference in Bayesian networks. In C. Boutilier & M. Goldszmidt (Eds.), *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference.* San Francisco: Morgan Kaufmann.

Jensen, F. V. (2001). *Bayesian networks and decision graphs.* New York: Springer.

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

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference.* San Mateo, CA: Morgan Kaufmann.

Takikawa, M., D'Ambrosio, B., & Wright, E. (2002). Realtime inference with large-scale temporal Bayes nets. In A. Darwiche & N. Friedman (Eds.), *Uncertainty in Artificial Intelligence: Proceedings of the Eighteenth Conference.* San Francisco: Morgan Kaufmann.

[3]http://www.coli.uni-sb.de/sfb378/
[4]http://w5.cs.uni-sb.de/∼ready