A Method for Implementing a Probabilistic Model as a Relational Database

S.K.M. Wong, C.J. Butz, and Y. Xiang
Department of Computer Science
University of Regina
Regina, Saskatchewan
Canada, S4S 0A2
< {wong,butz,yxiang}@cs.uregina.ca>

Abstract

This paper discusses a method for implementing a probabilistic inference system based on an extended relational data model. This model provides a unified approach for a variety of applications such as dynamic programming, solving sparse linear equations, and constraint propagation. In this framework, the probability model is represented as a generalized relational database. Subsequent probabilistic requests can be processed as standard relational queries. Conventional database management systems can be easily adopted for implementing such an approximate reasoning system.

1 Introduction

Probabilistic models [4, 9, 10] are used for making decisions under uncertainty. The input to a probabilistic model is usually a Bayesian network [10]. It may also consist of a set of potentials which define a Markov network [4]. In this paper, we assume that the probabilistic model is described by a Markov network. For this model, the propagation method [5, 6, 7, 12, 13] can be conveniently applied to convert the potentials into marginal distributions.

There is another important reason to characterize a probabilistic model by a Markov network, as it has been shown that such a network can be represented as a generalized relational database [14, 15, 16]. That is, the probabilistic model can be transformed into an equivalent (extended) relational data model. More specifically, the marginal corresponding to each potential can be viewed as a relation in the relational database. Furthermore, the database scheme derived from a Markov network forms an acyclic join dependency [15], which possesses many desirable properties [1, 8] in database applications.

As the probabilistic model is now represented by a relational data model, a probability request expressed as a conditional probability can be equivalently transformed into a standard query to be executed by the database management system. Naturally, all query optimization techniques can be directly applied to processing this query including data structure modification. Thus, these transformations allow us to take full advantage of the query optimizer and other performance enhancement capabilities available in traditional relational databases.

This paper, a sequel of the presentation in the IPMU conference [15], reports on the technical details involved in the design of a probabilistic inference system by transforming a Markov network into a relational database.

Our paper is organized as follows. In Section 2, for completeness we review a unified relational data model for both probabilistic reasoning and database management systems. In Section 3, we show that a factored probability distribution can be expressed as a generalized acyclic join dependency. The method for implementing a probabilistic inference system is described in Section 4. First, we describe how a relational database is constructed for a given probabilistic model. We then show that processing a request for evidential reasoning is equivalent to processing a standard relational query. We conclude by pointing out that the extended relational database system can in fact model a number of apparently different but closely related applications [12].

2 An Extended Relational Data Model for Probabilistic Inference

Before introducing our data model, we need to define some basic notions pertinent to our discussion such as: hypergraphs, factored distributions, and marginalization. Then we show how under certain conditions a factored joint probability distribution can be expressed
as a generalized acyclic join dependency in the extended relational model.

2.1 Basic Notions

Hypergraphs and Hypertrees:

Let \( \mathcal{L} \) denote a lattice. We say that \( \mathcal{H} \) is a hypergraph, if \( \mathcal{H} \) is a finite subset of \( \mathcal{L} \). Consider, for example, the power set \( 2^X \), where \( X = \{x_1, x_2, \ldots, x_n\} \) is a set of variables. The power set \( 2^X \) is a lattice of all subsets of \( X \). Any subset of \( 2^X \) is a hypergraph on \( 2^X \). We say that an element \( t \) in a hypergraph \( H \) is a twig if there exists another element \( b \) in \( \mathcal{H} \), distinct from \( t \), such that \( t \cap (\{H - \{t\}\}) = t \cap b \). We call any such \( b \) a branch for the twig \( t \). A hypergraph \( \mathcal{H} \) is a hypertree (an acyclic hypergraph [1]) if its elements can be ordered, say \( h_1, h_2, \ldots, h_n \), so that \( h_i \) is a twig in \( \{h_1, h_2, \ldots, h_i\} \), for \( i = 2, \ldots, n \). We call any such ordering a hypertree construction ordering for \( \mathcal{H} \). Given a hypertree construction ordering \( h_1, h_2, \ldots, h_n \), we can choose, for \( i \) from 2 to \( n \), an integer \( b(i) \) such that \( 1 \leq b(i) \leq i - 1 \) and \( h(i) \) is a branch for \( h_i \) in \( \{h_1, h_2, \ldots, h_i\} \). We call the function \( b(i) \) satisfying this condition a branching function for \( \mathcal{H} \) and \( h_1, h_2, \ldots, h_n \).

For example, let \( X = \{x_1, x_2, \ldots, x_6\} \) and \( \mathcal{L} = 2^X \). Consider a hypergraph \( \mathcal{H} = \{h_1 = \{x_1, x_2, x_3\}, h_2 = \{x_1, x_2, x_4\}, h_3 = \{x_3, x_4, x_5\}, h_4 = \{x_5, x_6\}\} \), depicted in Figure 1. This hypergraph is in fact a hypertree cover of the hypergraph, \( \{\{x_1, x_2\}, \{x_1, x_3\}, \{x_1, x_2, x_4\}, \{x_2, x_4\}, \{x_3, x_5\}, \{x_5, x_6\}\} \).

Factored Probability Distributions:

Let \( X = \{x_1, x_2, \ldots, x_n\} \) denote a set of variables. A factored probability distribution \( p(x_1, x_2, \ldots, x_n) \) can be written as:

\[
p(x_1, x_2, \ldots, x_n) = \phi = \phi_{h_1}\phi_{h_2}\cdots\phi_{h_n},
\]

where each \( h_i \) is a subset of \( X \), i.e., \( h_i \in 2^X \), and \( \phi_{h_i} \) is a real-valued function on \( h_i \). Moreover, \( X = h_1 \cup h_2 \cup \cdots \cup h_n = \bigcup_{i=1}^n h_i \). By definition, \( \mathcal{H} = \{h_1, h_2, \ldots, h_n\} \) is a hypergraph on the lattice \( 2^X \). Thus, a factored probability distribution can be viewed as a product on a hypergraph \( \mathcal{H} \), namely:

\[
\phi = \prod_{h \in \mathcal{H}} \phi_h.
\]

Let \( v_x \) denote the discrete frame (state space) of the variable \( x \in X \). We call an element of \( v_x \) a configuration of \( x \). We define \( v_h \) to be the Cartesian product of the frames of the variables in a hyperedge \( h \in 2^X \):

\[
v_h = \times_{x \in h} v_x.
\]

We call \( v_h \) the frame of \( h \), and we call its elements configurations of \( h \).

Let \( h, k \in 2^X \), and \( h \subseteq k \). If \( c \) is a configuration of \( k \), i.e., \( c \in v_k \), we write \( c \upharpoonright h \) for the configuration of \( h \) obtained by deleting the values of the variables in \( k \) and not in \( h \). For example, let \( h = \{x_1, x_2\}, k = \{x_1, x_2, x_3, x_4\} \), and \( c = (c_1, c_2, c_3, c_4) \), where \( c_i \in v_{x_i} \). Then, \( c \upharpoonright h = (c_1, c_2) \).

If \( h \) and \( k \) are disjoint subsets of \( X \), \( c_h \) is a configuration of \( h \), and \( c_k \) is a configuration of \( k \), then we write \( (c_h \circ c_k) \) for the configuration of \( h \cup k \) obtained by concatenating \( c_h \) and \( c_k \). In other words, \( (c_h \circ c_k) \) is the unique configuration of \( h \cup k \) such that \( (c_h \circ c_k) \upharpoonright h = c_h \) and \( (c_h \circ c_k) \upharpoonright k = c_k \). Using the above notation, a factored probability distribution \( \phi \) on \( \mathcal{H} \) can be defined as follows:

\[
\phi(c) = \left( \prod_{h \in \mathcal{H}} \phi_h \right)(c) = \prod_{h \in \mathcal{H}} \phi_h(c \upharpoonright h),
\]

where \( c \in v_X \) is an arbitrary configuration and \( X = \bigcup \mathcal{H} \).

Marginalization:
We can conveniently express each function $\phi_h$ of $x_1, x_2, \ldots, x_t$ by a relation $\Phi_h$. Suppose $h = \{x_1, x_2, \ldots, x_i\}$. The function $\phi_h$ can be expressed as a relation on the set $\{x_1, x_2, \ldots, x_i, f_{\phi_h}\}$ of attributes as shown in Figure 2. A configuration $c_i = (c_{i1}, c_{i2}, \ldots, c_{in})$ in the above table denotes a row excluding the last element in the row, and $s$ is the cardinality of $v_{\text{h}}$.

By definition, the product $\phi_h \cdot \phi_k$ of any two function $\phi_h$ and $\phi_k$ is given by:

$$(\phi_h \cdot \phi_k)(c) = \phi_h(c^{lh}) \cdot \phi_k(c^{lk}),$$

where $c \in v_{h \cup k}$. We can therefore express the product $\phi_h \cdot \phi_k$ equivalently as a product join of the relations $\Phi_h$ and $\Phi_k$, written $\Phi_h \bowtie \Phi_k$, which is defined as follows:

(i) Compute the natural join, $\Phi_h \bowtie \Phi_k$, of the two relations $\Phi_h$ and $\Phi_k$.

(ii) Add a new column with attribute $f_{\phi_h \cdot \phi_k}$ to the relation $\Phi_h \bowtie \Phi_k$ on $h \cup k$. Each value of $f_{\phi_h \cdot \phi_k}$ is given by $\phi_h(c^{lh}) \cdot \phi_k(c^{lk})$, where $c \in v_{h \cup k}$.

(iii) Obtain the resultant relation $\Phi_h \bowtie \Phi_k$ by projecting the relation obtained in Step (ii) on the set of attributes $h \cup k \cup \{f_{\phi_h \cdot \phi_k}\}$.

For example, let $h = \{x_1, x_2\}$, $k = \{x_2, x_3\}$, and $v_h = v_k = \{0, 1\}$. The product join $\Phi_h \bowtie \Phi_k$ is illustrated in Figure 3.

\[\begin{array}{ccc}
\hline
x_1 & x_2 & f_{\phi_h} \\
\hline
0 & 0 & a_1 \\
0 & 1 & a_2 \\
1 & 0 & a_3 \\
1 & 1 & a_4 \\
\hline
\end{array}\]

\[\begin{array}{ccc}
\hline
x_3 & f_{\phi_k} \\
\hline
0 & b_1 \\
0 & 1 & b_2 \\
1 & 0 & b_3 \\
1 & 1 & b_4 \\
\hline
\end{array}\]

\[\begin{array}{ccc}
\hline
x_1 & x_2 & x_3 & f_{\phi_h \cdot \phi_k} \\
\hline
0 & 0 & 0 & a_1 \cdot b_1 \\
0 & 0 & 1 & a_1 \cdot b_2 \\
0 & 1 & 0 & a_2 \cdot b_3 \\
0 & 1 & 1 & a_2 \cdot b_4 \\
1 & 0 & 0 & a_3 \cdot b_3 \\
1 & 0 & 1 & a_3 \cdot b_4 \\
1 & 1 & 0 & b_3 \\
1 & 1 & 1 & b_4 \\
\hline
\end{array}\]
consider, for example, the relation \( t \) of marginalization.

Before discussing the computation of marginals of a relation \( t \) with attributes \( x_1, x_2, x_3, f_{\Phi_k} \), and \( k = \{x_1, x_2, x_3\} \).

**Lemma 1** [12, 15]

(i) If \( \Phi_k \) is a relation on \( k \), and \( h \subseteq g \subseteq k \), then

\[
(\Phi_k^g)^h = \Phi_k^h.
\]

(ii) If \( \Phi_h \) and \( \Phi_k \) are relations on \( h \) and \( k \), respectively, then

\[
(\Phi_h \otimes \Phi_k)^h = \Phi_h \otimes \left(\Phi_k^{h \cap k}\right).
\]

Before discussing the computation of marginals of a factored distribution, let us first state the notion of computational feasibility introduced by Shafer [12]. We call a set of attributes feasible if it is feasible to represent relations on these attributes, join them, and marginalize on them. We assume that any subset of feasible attributes is also feasible. Furthermore, we assume that the factored distribution is represented on a hypertree and every element in \( \Phi \) is feasible.

**Lemma 2** [12, 15] Let \( \Phi = \otimes \{\Phi_h|h \in \mathcal{H}\} \) be a factored probability distribution on a hypertree \( \mathcal{H} \). Let \( t \) be a twig in \( \mathcal{H} \) and \( b \) be a branch for \( t \). Then,

(i) \[
(\otimes \{\Phi_h|h \in \mathcal{H}\})^{i \cup \mathcal{H}^{-i}} = (\otimes \{\Phi_h|h \in \mathcal{H}^{-i}\}) \otimes \Phi_{i \cap b}^b.
\]

(ii) If \( k \subseteq \mathcal{H}^{-i} \), then \[
(\otimes \{\Phi_h|h \in \mathcal{H}\})^{i \cup k} = (\otimes \{\Phi_h^{-i}|h \in \mathcal{H}^{-i}\})^{i \cup k},
\]

where \( \mathcal{H}^{-i} \) denotes the set of hyperedges \( \mathcal{H} - \{t\} \), \( \Phi_{i \cap b}^b = \Phi_b \otimes \Phi_{i \cap b}^b \), and \( \Phi_{i \cap b}^b = \Phi_b \) for all other \( h \) in \( \mathcal{H}^{-i} \).

We now describe a procedure for computing \( \Phi_{i \cap b}^b \) for \( \Phi_b \in \mathcal{H} \), where \( \Phi = \otimes \{\Phi_h|h \in \mathcal{H}\} \) and \( \mathcal{H} \) is a hypertree. Choose a hypertree construction ordering for \( \mathcal{H} \) that begins with \( h_1 = k \) as the root, say \( h_1, h_2, ..., h_n \), and choose a branching \( b(i) \) function for this particular ordering. For \( i = 1, 2, ..., n \), let

\[
\mathcal{H}^i = \{h_1, h_2, ..., h_i\}.
\]

This is a sequence of sub-hypertrees, each larger than the last; \( \mathcal{H}^1 = \{h_1\} \) and \( \mathcal{H}^n = \mathcal{H} \). The element \( h_i \) is a twig in \( \mathcal{H}^i \). To compute \( \Phi_{i \cap b}^b \), we start with \( \Phi_{i \cap b}^b \) going backwards in this sequence. We use Lemma 2 each time to perform the reduction. At the step from \( \mathcal{H}^i \) to \( \mathcal{H}^{i-1} \), we go from \( \Phi_{i \cup \mathcal{H}^{i-1}} \) to \( \Phi_{i \cup \mathcal{H}^{i-1}} \). We omit \( h_i \) in \( \mathcal{H}^i \) and change the relation on \( h_{b(i)} \) in \( \mathcal{H}^{i-1} \) from \( \Phi_{h_{b(i)}}^b \) to

\[
\Phi_{h_{b(i)}}^{i-1} = \Phi_{h_{b(i)}}^b \otimes (\Phi_{h_{b(i)}}^i)^{i \cup i \cap h_{b(i)}},
\]

and the other relations in \( \mathcal{H}^{i-1} \) are not changed. The collection of relations with which we begin, \( \{\Phi_h|h \in \mathcal{H}\} \), is simply \( \{\Phi_h|h \in \mathcal{H}\} \), and the collection with which we end, \( \{\Phi_h|h \in \mathcal{H}^1\} \), consists of the single relation \( \Phi_1 = \Phi_{i \cap h_1} \).

Consider a factored probability distribution \( \Phi = \otimes \{\Phi_h|h \in \mathcal{H}\} \) on a hypertree \( \mathcal{H} = \{h_1, h_2, ..., h_n\} \). We say that \( \Phi \) satisfies the acyclic join dependency (AUDJ), \(*\{h_1, h_2, ..., h_n\}, \) if \( \Phi \) decomposes losslessly onto a hypertree construction ordering \( h_1, h_2, ..., h_n \), i.e., \( \Phi \) can be expressed as:

\[
\Phi = (...((\Phi_{h_1} \otimes \Phi_{h_2}) \otimes \Phi_{h_3})... \otimes \Phi_{h_n}),
\]

where \( \otimes \) is a generalized join operator defined by:

\[
\Phi_{i \cap j} \otimes \Phi_{j \cap k} = \Phi_{i \cap j} \otimes \Phi_{j \cap k} \otimes (\Phi_{i \cap j \cap k})^{-1}.
\]
The relation \((\Phi^{lh})^{-1}\) is defined as follows. First, let us define the inverse function \((\phi^{lh})^{-1}\) of \(\phi\). That is,

\[
(\phi^{lh})^{-1}(c) = \left(\sum_{c'} \phi(c \circ c')\right)^{-1}, \text{ where } \sum_{c'} \phi(c \circ c') > 0,
\]

c is a configuration of \(h \subseteq \mathcal{H}\), and \(c'\) is a configuration of \(\mathcal{H} - h\). We call the function \((\phi^{lh})^{-1}\) the inverse marginal of \(\phi\) on \(h\). The inverse relation \((\Phi^{lh})^{-1}\) is the relation constructed from the inverse function \((\phi^{lh})^{-1}\). Obviously, the product \((\Phi^{lh})^{-1} \circ \Phi^{lh}\) is a unit function on \(h\), and \((\Phi^{lh})^{-1} \circ \Phi^{lh}\) is an identity relation on \(h\).

**Theorem 1** [15] Any factored probability distribution \(\Phi = \bigotimes \{\Phi_h | h \in \mathcal{H}\}\) on a hypertree, \(\mathcal{H} = \{h_1, h_2, ..., h_n\}\), decomposes losslessly onto a hypertree construction ordering \(h_1, h_2, ..., h_n\). That is, \(\Phi\) satisfies the AJD, \(*[h_1, h_2, ..., h_n]*\).

**Proof:**

Suppose \(t \in \mathcal{H}\) is a twig. By Lemma 2,

\[
\Phi^{l\mathcal{H}^{-t}} = ((\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\}) \circ \Phi_t)^{l\mathcal{H}^{-t}} = ((\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\}) \circ \Phi_t^{l\mathcal{H}^{-t}}) \circ \Phi_t = ((\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\}) \circ \Phi_t^{l\mathcal{H}^{-t}}) \circ \Phi_t.
\]

Note that \((\Phi_t^{l\mathcal{H}^{-t}})^{-1} \circ \Phi_t^{l\mathcal{H}^{-t}}\) is an identity relation on \(t \cap b\). Thus,

\[
\Phi^{l\mathcal{H}^{-t}} \circ (\Phi_t^{l\mathcal{H}^{-t}})^{-1} \circ \Phi_t = (\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\}) \circ \Phi_t^{l\mathcal{H}^{-t}} \circ (\Phi_t^{l\mathcal{H}^{-t}})^{-1} \circ \Phi_t = (\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\}) \circ \Phi_t = \Phi.
\]

Now we want to show that:

\[
(\Phi_t^{l\mathcal{H}^{-t}})^{-1} \circ \Phi_t = (\Phi_t^{l\mathcal{H}^{-t}})^{-1} \circ \Phi_t.
\]

Note that by property (ii) of Lemma 1, we obtain:

\[
\Phi_t \circ (\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\})^{l\mathcal{H}^{-t}} = (\Phi_t \circ (\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\}))^{l\mathcal{H}^{-t}} = \Phi_t^{l\mathcal{H}^{-t}}.
\]

On the other hand, we have:

\[
(\Phi_t^{l\mathcal{H}^{-t}})^{-1} \circ ((\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\})^{l\mathcal{H}^{-t}})^{-1} = ((\Phi_t \circ (\bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\}))^{l\mathcal{H}^{-t}})^{-1} = (\Phi_t^{l\mathcal{H}^{-t}})^{-1} = (\Phi_t^{l\mathcal{H}^{-t}})^{-1}.
\]

Hence,

\[
\Phi_t \circ (\Phi_t^{l\mathcal{H}^{-t}})^{-1} = \Phi_t \circ (\Phi_t^{l\mathcal{H}^{-t}})^{-1}.
\]

The relation \(\Phi\) can therefore be expressed as:

\[
\Phi = \Phi^{l\mathcal{H}^{-t}} \circ \Phi_t \circ (\Phi_t^{l\mathcal{H}^{-t}})^{-1} = \Phi^{l\mathcal{H}^{-t}} \circ \Phi_t^{l\mathcal{H}^{-t}}.
\]

Moreover,

\[
\Phi^{l\mathcal{H}^{-t}} = \bigotimes \{\Phi_h | h \in \mathcal{H}^{-t}\} \circ \Phi_t^{l\mathcal{H}^{-t}} = \bigotimes \{\Phi_h^{-t} | h \in \mathcal{H}^{-t}\}.
\]

We can immediately apply the same procedure to \(\Phi^{l\mathcal{H}^{-t}}\) for further reduction. Thus, by applying this algorithm recursively, the desired result is obtained. \(\square\)

### 4 The method for implementing a Probabilistic Inference System

In order to convert a probabilistic model into a relational model, first we need to be able to efficiently transform the input potentials into marginals. Since we assume that the hypergraph induced by the potentials is a hypertree, we can apply the propagation method \([6, 12]\) to compute all their marginals. This process involves first moving backward along the hypertree construction ordering to find the marginal of the root, then moving forward from the root to the leaves for determining marginals of the other potentials.

The next task is to transform a probability request into a standard relational query addressed to the database which is equivalent to the original probability model. The relational query can be formulated by scanning the probability request to determine the marginals involved along the hypertree construction ordering, as well as the specific variables (attributes) within each respective marginal. Once the query is expressed in terms of the query language provided, it is then submitted to and processed by the standard database management system in the usual manner.

### 4.1 Transformation of Potentials to Marginals (Relations)

We are given as input a set of potentials \(\phi_h\)'s which define a factored joint probability distribution \(\Phi = \bigotimes \{\Phi_h | h \in \mathcal{H}\}\), where \(\mathcal{H} = \{h_1, h_2, ..., h_n\}\) is the corresponding hypergraph. The first step in this transformation is to check if the hypergraph \(\mathcal{H}\) is a hypertree \([1]\), but if so determine a branching function \(b(i)\) for it. If we do not have a hypertree, then some potentials...
can be combined so that the resultant hypergraph is a hypertree [12].

In the following discussion, we henceforth assume that \( \mathcal{H} = \{ h_1, h_2, \ldots, h_n \} \) is a hypertree. Let the branching function \( b(i), i = 2, \ldots, n \) define a hypertree construction ordering. The procedure for computing the marginal of the root \( h_1 \) by moving backward along the hypertree construction ordering has been described in Section 3.

Once we have determined the root marginal, \( \Phi^{i h_1} \), we may move forward along the hypertree construction ordering to compute marginals of the other potentials. For this purpose, while we are moving backward we should save the relation \( (\Phi^{i h_1})^{L h_1 h_i} \) at each stage \( \mathcal{H}' \). Then we can determine the other marginals by the formula:

\[
\Phi^{i h_1} = \Phi^{i h_1} \odot (\Phi^{i h_1})^{L h_1 h_i} \odot (\Phi^{i h_1})^{L h_1 h_i}.
\]

To see this, consider the situation where we have just computed the root marginal \( \Phi^{i h_1} \), namely: \( \Phi^{i h_1} = \Phi^{i h_1} \odot (\Phi^{i h_1})^{L h_1 h_i} \). Note that \( \Phi^{i 2} = (\Phi')^{i 2} \), where \( \Phi' = \otimes \{ h \mid h \in \mathcal{H}' \} \). By Lemma 1, we obtain:

\[
(\Phi^{i h_1})^{L h_1 h_i} = (\Phi^{i h_1} \odot (\Phi^{i h_1})^{L h_1 h_i})^{L h_1 h_i}.
\]

From Lemmas 1 and 2, it follows:

\[
\Phi^{i h_2} \odot (\Phi^{i h_1})^{L h_1 h_i} = (\Phi^{i h_1} \odot (\Phi^{i h_1})^{L h_1 h_i})^{L h_1 h_i}.
\]

Hence, by continuing moving forward, we will arrive at the above general formula.

Consider, for example, a factored joint probability distribution defined by six potentials [4] as shown in column 2 of Tables 1 to 6. We have modified the column names to reflect the notation used in this paper. The corresponding hypergraph, \( \mathcal{H} = \{ h_1 = \{ x_1, x_2 \}, h_2 = \{ x_2, x_3, x_4, x_5 \}, h_3 = \{ x_2, x_3, x_4, x_5 \}, h_4 = \{ x_2, x_3, x_4, x_5 \}, h_5 = \{ x_2, x_3, x_4, x_5 \}, h_6 = \{ x_2, x_3, x_4, x_5 \}, \} \) is depicted in Figure 7. This hypergraph is in fact a hypertree. The sequence \( h_1, h_2, h_3, h_4, h_5, h_6 \) is a hypertree construction ordering which defines the branching function, \( b(2) = 1, b(3) = 2, b(4) = 3, b(5) = 4, b(6) = 5 \).

To compute the root marginal \( \Phi^{i h_1} \), we may move backward from the leaf hyperedge towards the root \( h_1 \) along the hypertree construction ordering. Thus we first transform the hypergraph \( \mathcal{H}'(= \mathcal{H}) \) to \( \mathcal{H}' \). That

| configuration | \( \Phi^{i h_2} \) | \( \Phi^{i h_3} \) | \( \Phi^{i h_4} \) | \( \Phi^{i h_5} \) |
|--------------|----------------|----------------|----------------|----------------|
| \( -x_2 \)   | 0.502          | 0.391          | 0.391          |                |
| \( -x_2 \)   | 0.261          | 0.058          | 0.058          |                |
| \( x_1 \)    | 0.498          | 0.387          | 0.387          |                |
| \( x_2 \)    | 0.739          | 0.164          | 0.164          |                |

Table 1: \( h_1 = \{ x_1, x_2 \} \)

| configuration | \( \Phi^{i h_2} \) | \( \Phi^{i h_3} \) | \( \Phi^{i h_4} \) | \( \Phi^{i h_5} \) |
|--------------|----------------|----------------|----------------|----------------|
| \( -x_2 \)   | 0.435          | 0.431          | 0.335          |                |
| \( -x_2 \)   | 0.000          | 0.000          | 0.000          |                |
| \( -x_2 \)   | 0.000          | 0.000          | 0.000          |                |
| \( x_2 \)    | 0.435          | 0.450          | 0.100          |                |
| \( x_2 \)    | 0.029          | 0.122          | 0.027          |                |
| \( x_2 \)    | 0.061          | 0.212          | 0.047          |                |
| \( x_2 \)    | 0.029          | 0.009          | 0.002          |                |
| \( x_2 \)    | 0.739          | 0.063          | 0.014          |                |

Table 2: \( h_2 = \{ x_2, x_3, x_4, x_5 \} \)

is, we omit \( h_6 \) in \( \mathcal{H}' \) and change the relation \( \Phi_{h_5} \) in \( \mathcal{H}' \) to \( \Phi_{h_5} \) defined by:

\[
\Phi_{h_5} = \Phi_{h_5} \odot (\Phi_{h_5})^{L h_1 h_5}.
\]

and the other relations in \( \mathcal{H}' \) are not changed. Similarly we have:

\[
\Phi_{h_4} = \Phi_{h_4} \odot (\Phi_{h_4})^{L h_1 h_4},
\]

\[
\Phi_{h_2} = \Phi_{h_2} \odot (\Phi_{h_2})^{L h_1 h_2},
\]

\[
\Phi_{h_1} = \Phi_{h_1} \odot (\Phi_{h_1})^{L h_1 h_1}.
\]

As \( \Phi^{i h_1} = \Phi^{i h_1} \), we have thus determined the root marginal by moving backward. Now we start moving forward from the root. By applying the formula for computing other marginals, we immediately obtain:

\[
\Phi^{i h_2} = \Phi^{i h_2} \odot ((\Phi_{h_2})^{L h_2 h_2})^{-1} (\Phi_{h_2})^{L h_1 h_2},
\]

\[
\Phi^{i h_3} = \Phi^{i h_3} \odot ((\Phi_{h_3})^{L h_3 h_3})^{-1} (\Phi_{h_3})^{L h_1 h_3},
\]

\[
\Phi^{i h_4} = \Phi^{i h_4} \odot ((\Phi_{h_4})^{L h_4 h_4})^{-1} (\Phi_{h_4})^{L h_1 h_4},
\]

\[
\Phi^{i h_5} = \Phi^{i h_5} \odot ((\Phi_{h_5})^{L h_5 h_5})^{-1} (\Phi_{h_5})^{L h_1 h_5},
\]

\[
\Phi^{i h_6} = \Phi^{i h_6} \odot ((\Phi_{h_6})^{L h_6 h_6})^{-1} (\Phi_{h_6})^{L h_1 h_5}.
\]

The numerical results are shown in the last column of Tables 1 to 6. These relations \( \Phi^{i h_1} \) form an acyclic join dependency in our extended relational data model.
Table 3: \( h_3 = \{x_2, x_4, x_5, x_6\} \)

| configuration | \( \phi(x, z_2, z_5, x_6) \) | \( \Phi_{h_3}^* \otimes ((\Phi_{h_3}^*)^{t_{h_3} \cap h_2})^{-1} \) | \( \Phi_{h_2} \) |
|---------------|---------------------|-------------------------------------------------|---------------------|
| \( x_2 = x_4 \) | 0.561               | 0.562                                           | 0.267               |
| \( x_2 = x_5 \) | 0.371               | 0.398                                           | 0.176               |
| \( x_2 = x_5 \) | 0.519               | 0.676                                           | 0.226               |
| \( x_2 = x_5 \) | 0.250               | 0.324                                           | 0.109               |
| \( x_3 = x_4 \) | 0.016               | 0.235                                           | 0.000               |
| \( x_3 = x_5 \) | 0.022               | 0.765                                           | 0.000               |
| \( x_2 = x_4 \) | 0.057               | 0.561                                           | 0.018               |
| \( x_2 = x_5 \) | 0.371               | 0.398                                           | 0.011               |
| \( x_3 = x_5 \) | 0.016               | 0.765                                           | 0.001               |
| \( x_2 = x_5 \) | 0.250               | 0.325                                           | 0.015               |
| \( x_2 = x_5 \) | 0.250               | 0.235                                           | 0.008               |
| \( x_2 = x_5 \) | 0.052               | 0.765                                           | 0.026               |
| \( x_2 = x_5 \) | 0.052               | 0.765                                           | 0.029               |
| \( x_2 = x_5 \) | 0.373               | 0.749                                           | 0.085               |

Table 4: \( h_4 = \{x_2, x_5, x_6\} \)

| configuration | \( \phi(x, z_2, z_7) \) | \( \Phi_{h_4}^* \otimes ((\Phi_{h_4}^*)^{t_{h_4} \cap h_3})^{-1} \) | \( \Phi_{h_3} \) |
|---------------|---------------------|-------------------------------------------------|---------------------|
| \( x_2 = x_7 \) | 0.579               | 0.579                                           | 0.285               |
| \( x_2 = x_7 \) | 0.421               | 0.421                                           | 0.208               |
| \( x_2 = x_7 \) | 0.563               | 0.563                                           | 0.160               |
| \( x_2 = x_7 \) | 0.437               | 0.437                                           | 0.125               |
| \( x_2 = x_7 \) | 0.579               | 0.579                                           | 0.049               |
| \( x_2 = x_7 \) | 0.421               | 0.421                                           | 0.036               |
| \( x_2 = x_7 \) | 0.563               | 0.563                                           | 0.077               |
| \( x_2 = x_7 \) | 0.437               | 0.437                                           | 0.060               |

Table 5: \( h_5 = \{x_2, x_7, x_8\} \)

| configuration | \( \phi(x, z_2, z_7) \) | \( \Phi_{h_5}^* \otimes ((\Phi_{h_5}^*)^{t_{h_5} \cap h_4})^{-1} \) | \( \Phi_{h_4} \) |
|---------------|---------------------|-------------------------------------------------|---------------------|
| \( x_2 = x_7 \) | 0.597               | 0.597                                           | 0.310               |
| \( x_2 = x_7 \) | 0.303               | 0.303                                           | 0.135               |
| \( x_2 = x_7 \) | 0.722               | 0.722                                           | 0.240               |
| \( x_2 = x_7 \) | 0.278               | 0.278                                           | 0.093               |
| \( x_2 = x_7 \) | 0.433               | 0.433                                           | 0.055               |
| \( x_2 = x_7 \) | 0.567               | 0.567                                           | 0.071               |
| \( x_2 = x_7 \) | 0.261               | 0.261                                           | 0.025               |
| \( x_2 = x_7 \) | 0.729               | 0.729                                           | 0.071               |

Table 6: \( h_6 = \{x_7, x_8, x_9\} \)

| configuration | \( \phi(x, z_7, z_9) \) | \( \Phi_{h_6}^* \otimes ((\Phi_{h_6}^*)^{t_{h_6} \cap h_5})^{-1} \) | \( \Phi_{h_5} \) |
|---------------|---------------------|-------------------------------------------------|---------------------|
| \( x_7 = x_9 \) | 0.647               | 0.647                                           | 0.236               |
| \( x_7 = x_9 \) | 0.353               | 0.353                                           | 0.129               |
| \( x_7 = x_9 \) | 0.579               | 0.579                                           | 0.119               |
| \( x_7 = x_9 \) | 0.421               | 0.421                                           | 0.088               |
| \( x_7 = x_9 \) | 0.594               | 0.594                                           | 0.157               |
| \( x_7 = x_9 \) | 0.406               | 0.406                                           | 0.108               |
| \( x_7 = x_9 \) | 0.438               | 0.438                                           | 0.072               |
| \( x_7 = x_9 \) | 0.562               | 0.562                                           | 0.092               |

4.2 Transformation of a Probability Request to a Query

Just as we can transform a potential \( \Phi_{h_5} \), to a marginal relation \( \Phi^{h_5} \), we can transform a probability request of the form \( p(x_2, \ldots, x_9|x_e = \epsilon, \ldots, x_g = \gamma) \) to a relational query. This query can then be processed by the database management system. There are, however, two ways to construct the query depending on whether the product join (\( \otimes \)) and generalized join (\( \otimes' \)) operators have been incorporated into the database management system. We will show how to transform the probability request to a relational query in either situation.

(i) In the first case we assume that the database management system has been extended to include the product join and generalized join operators. Then with respect to a particular hypertree construction ordering, we first determine the join-path \( h_r, \ldots, h_s \) such that the union \( h_r \cup \ldots \cup h_s \) of these relation schemes (hyperedges) contains all the variables in the probability request \( p(x_2, \ldots, x_9|x_e = \epsilon, \ldots, x_g = \gamma) \). Then the relation \( \Phi_x = \Phi_{\{x_2, \ldots, x_9\}} \), depicted in Figure 7, is being constructed by the query:

SELECT \( x_2, \ldots, x_d \)
INTO \( \Phi_x \)
FROM \( \Phi^{h_r} \otimes' \ldots \otimes' \Phi^{h_s} \)
WHERE \( x_e = \epsilon, \ldots, x_g = \gamma \).

At this point, we have the information needed to answer the probability request all in a single relation \( \Phi_x \). However, to compute the required conditional probability, we need to marginalize \( \Phi_x \) onto \( \{x_2, \ldots, x_d\} \) by the following query:

SELECT \( x_2, \ldots, x_d \)
SUM (\( f_{\phi_x} \))
INTO \( \Psi_x \)
FROM \( \Phi_x \)
GROUPBY \( x_2, \ldots, x_d \)

Since the relation \( \Psi_x \) is not normalized, we have to define the normalization relation \( \Psi_x \) which is a constant relation as shown in Figure 8, where
A Method for Implementing a Probabilistic Model as a Relational Database

The constant relation $\Psi_k$.

Finally, the answer to the probability request is given by the relation $\Psi_k \otimes \Psi_k^{-1}$. This demonstrates that any probability request can be easily answered by submitting simple queries as described to the relational database management system.

The above discussion indicates that we need not implement the marginalization operator down, as the standard relational query languages already provide the SUM and GROUP BY facilities. These two functions are indeed equivalent to the marginalization operation.

(ii) In the second case we simulate the product join and generalized join operators as we are interfacing with a standard database management system. We will first discuss the simulation of the product join $\otimes$ and generalized join $\otimes'$ operators, before we construct the relation to answer the probability request.

Suppose we want to compute the product join of two relations $\Phi_h$ and $\Phi_k$, i.e., $\Phi_h \otimes \Phi_k$. According to the definition of $\otimes$ (see the example in Figure 3), we construct the relation $\Phi_h \otimes \Phi_k$ by the query:

```
SELECT h \cup k, f_{\phi_h}, f_{\phi_k}
INTO \Phi_{h\otimes k}
FROM \Phi_h, \Phi_k.
```

Next we create a new column labelled by the attribute $f_{\phi_h \cdot \phi_k}$, representing the product $\phi_h \cdot \phi_k$ by the query:

```
ALTER TABLE \Phi_{h\otimes k} ADD f_{\phi_h \cdot \phi_k} FLOAT.
```

By definition, the entries in this column are:

$$(\phi_h \cdot \phi_k)(c) = \phi_h(c^{\phi_h}) \cdot \phi_k(c^{\phi_k}),$$

where $c \in v_{h\otimes k}$. The following query:

```
UPDATE \Phi_{h\otimes k}
SET f_{\phi_h \cdot \phi_k} = f_{\phi_h} \cdot f_{\phi_k},
```

accomplishes this task. The last step in simulating the product join $\otimes$ is to project $\Phi_{h\otimes k}$ onto the set of attributes $h \cup k \cup \{f_{\phi_h \cdot \phi_k}\}$ using the query:

```
SELECT h \cup k, f_{\phi_h}, f_{\phi_k}
INTO \Phi_{h\otimes k}
FROM \Phi_{h\otimes k}.
```

Thus we have derived the relation $\Phi_{h\otimes k} = \Phi_h \otimes \Phi_k$.

Since $\Phi_h \otimes' \Phi_k = \Phi_h \otimes \Phi_k \otimes \Phi_{h\cap k}^{-1}$, the simulation of the generalized join $\otimes'$ is just a simple extension of the product join $\otimes$. That is we need only compute $\Phi_{h\cap k}^{-1}$, the inverse relation of $\Phi_{h\cap k}$. We construct $\Phi_{h\cap k}$ by the query:

```
SELECT h \cap k, SUM(f_{\phi_h})
INTO \Phi_{h\cap k}
FROM \Phi_h
GROUPBY h \cap k.
```

Note that we can use $SUM(f_{\phi_h})$ and $\Phi_k$ in the above query, since $\Phi_{h\cap k}^{-1} = \Phi_{h\cap k}$. It is straightforward to construct the inverse relation $\Phi_{h\cap k}^{-1}$ from $\Phi_{h\cap k}$. Now the relation $\Phi_{h\otimes' k} = \Phi_h \otimes' \Phi_k$ is obtained by performing the product join $\Phi_{h\otimes k} \otimes \Phi_{h\cap k}^{-1}$.

Let $h_r, h_{r+1}, \ldots, h_{s-1}, h_s$ denote the join-path. We can compute the relation $\Psi_k = ((\ldots((\Phi_{h_r} \otimes' \Phi_{h_{r+1}}) \otimes' \ldots) \otimes' \Phi_{h_{s-1}}) \otimes' \Phi_{h_s})$ by repeatedly applying the generalized join operation. It is understood that the selection $x_r = \epsilon, \ldots, x_s = \gamma$ has been performed on each of the relations in the join-path before $\Psi_k$ is computed.

The relation $\Psi_k$, depicted in Figure 7, is obtained by the query:

```
SELECT x_1, \ldots, x_d
INTO \Psi_k
FROM \Psi_{h\cap k}.
```

We construct $\Psi_k$, and $\tilde{\Psi}_k$, depicted in Figure 8, as described in (i) of this subsection. The relation $\Psi_k \otimes \tilde{\Psi}_k^{-1}$ is the answer to the given probability request.

5 Conclusion

Once it is acknowledged that a probabilistic model can be viewed as an extended relational data model, it immediately follows that a probabilistic model can be implemented as an everyday database application. Thus, we are spared the arduous task of designing and
implementing our own probabilistic inference system and the associated costs. Even if such a system was successfully implemented, the resulting performance may not be comparable to that of existing relational databases. Our approach enables us to take advantage of the various performance enhancement techniques including query processing, query optimization, and data structure storage and manipulation, available in traditional relational database management systems. Thus the time required for belief update and answering probability requests is shortened.

The proposed relational data model also provides a unified approach to design both database and probabilistic reasoning systems.

In this paper, we have defined the product join operator $\otimes$ based on ordinary multiplication primarily because we are dealing with probabilities. By defining $\otimes$ differently (e.g. based on addition), our relational data model can be easily extended to solve a number of apparently different but closely related problems such as dynamic programming [2], solving sparse linear equations [11], and constraint propagation [3].

References

[1] C. Beeri, R. Fagin, D. Maier and M. Yannakakis, “On the desirability of acyclic database schemes,” Journal of the Association for Computing Machinery, vol. 30, 479-513, 1983.

[2] U. Bertelè and F. Brioschi, Nonserial Dynamic Programming. Academic Press, 1972.

[3] R. Dechter, A. Dechter and J. Pearl, “Optimization in constraint networks,” in Influence Diagrams, Belief Nets, and Decision Analysis, edited by R.M. Oliver and J.Q. Smith, Wiley, 1990.

[4] P. Hajek, T. Havranek and R. Jirousek, Uncertain Information Processing in Expert Systems. CRC Press, 1992.

[5] F.V. Jensen, “Junction tree and decomposable hypergraphs,” Technical report, JUDEX, Aalborg, Denmark, 1988.

[6] F.V. Jensen, S.L. Lauritzen, and K.G. Olesen, “Bayesian updating in causal probabilistic networks by local computations,” Computational Statistics Quarterly, vol. 4, 269-282, 1990.

[7] S.L. Lauritzen and D.J. Spiegelhalter, “Local computation with probabilities on graphical structures and their application to expert systems,” Journal of the Royal Statistical Society, Series B, vol 50, 157-244, 1988.

[8] D. Maier, The Theory of Relational Databases. Computer Science Press, 1983.

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

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

[11] D.J. Rose, “Triangulated graphs and the elimination process,” Journal of Mathematical Analysis and Its Applications, 32, 597-609, 1970.

[12] G. Shafer, “An axiomatic study of computation in hypertrees,” School of Business Working Paper Series, (No. 232), University of Kansas, Lawrence, 1991.

[13] P. Shenoy, “Valuation-based systems for discrete optimization,” Proc. Sixth Conference on Uncertainty in Artificial Intelligence, 334-335, 1990.

[14] S.K.M. Wong, “An extended relational data model for probabilistic reasoning,” submitted to publication, 1994.

[15] S.K.M. Wong, Y. Xiang and X. Nie, “Representation of bayesian networks as relational databases,” Proc. Fifth International Conference Information Processing and Management of Uncertainty in Knowledge-based systems, 159-165, 1994.

[16] S.K.M. Wong, Z.W. Wang, “On axiomatization of probabilistic conditional independence,” Proc. Tenth Conference on Uncertainty in Artificial Intelligence, 591-597, 1994.