Answering Queries from Context-Sensitive Probabilistic Knowledge Bases*

Liem Ngo and Peter Haddawy
Department of Electrical Engineering and Computer Science
University of Wisconsin-Milwaukee
PO Box 784
Milwaukee, WI 53201
{*liem, haddawy*}*@cs.uwm.edu*
414 229-4955

March 15, 1995

Abstract

We define a language for representing context-sensitive probabilistic knowledge. A knowledge base consists of a set of universally quantified probability sentences that include context constraints, which allow inference to be focused on only the relevant portions of the probabilistic knowledge. We provide a declarative semantics for our language. We present a query answering procedure which takes a query Q and a set of evidence E and constructs a Bayesian network to compute $P(Q|E)$. The posterior probability is then computed using any of a number of Bayesian network inference algorithms. We use the declarative semantics to prove the query procedure sound and complete. We use concepts from logic programming to justify our approach.

Keywords: reasoning under uncertainty, Bayesian networks, Probability model construction, logic programming

Submitted to *Theoretical Computer Science* special issue on Uncertainty in Databases and Deductive Systems.

*This work was partially supported by NSF grant IRI-9207262.

1 Introduction

Bayesian networks [14] have become the most popular method for representing and reasoning with probabilistic information [10]. But the complexity of inference in Bayesian networks [4] limits the size of the models that can be effectively reasoned over. Recently the approach known as knowledge-based model construction [17] has attempted to address this limitation by representing probabilistic information in a knowledge base using schematic variables and indexing schemes and constructing a network model tailored to each specific problem. The constructed network is a subset of the domain model represented by the collection of sentences in the knowledge base. Approaches to achieving this functionality have either focused on practical representations and model construction algorithms, neglecting formal aspects of the problem [8, 3], or they have focused on formal aspects of the knowledge base representation language, with less concern for ease of use of the representation and practical algorithms for constructing networks [15, 2]. Hence none of this work has provided a practical network construction algorithm which is proven correct.

We take a more encompassing approach to the problem by presenting a natural representation language with a rigorous semantics together with a network construction algorithm. In previous work [9] we started to address this problem by presenting a framework for constructing Bayesian networks from knowledge bases of first-order probability logic sentences. But the language used in that work was too constrained to be applicable to many practical problems. In this paper, we define a class of languages that can be used to describe a large class of probabilistic knowledge bases. A knowledge base consists of a set of universally quantified sentences of the form $(P(\text{consequent} \mid \text{antecedent}) = \alpha) \leftarrow \text{context}$. For a given probabilistic knowledge base, its declarative semantics is defined without reference to any particular reasoning procedure. We present a Bayesian network construction algorithm whose generated networks give correct answers to queries. We use related concepts in logic programming to justify our approach.

When reasoning with a large knowledge base, context information is necessary in order to focus attention on only the relevant portions. Consider the following motivating example, which will be referred to throughout the remainder of the paper. A burglary alarm could be triggered by a burglary, an earthquake, or a tornado. The likelihood of a burglary is influenced by the type of neighborhood one resides in. In different contexts, people might have different beliefs concerning the probabilistic relations between these variables. Someone who resides in California might think of an earthquake when he hears an alarm. If he lived in Wisconsin instead, he might relate that event with a possible tornado. Two networks representing these two different contexts are shown in Figure 1. Even for two people in Wisconsin, if one had been burglarized in the past he might associate a stronger weight with the relation between burglary and alarm. In general, causal probabilistic relationships are highly dependent upon context. Without context, it might be very difficult to glean from an expert concrete answers about his beliefs.

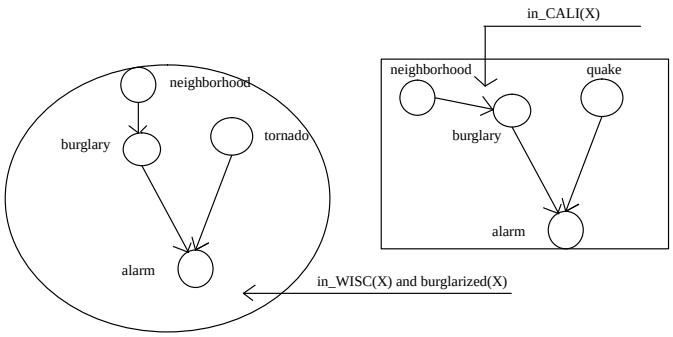

Figure 1: An example of context-dependent beliefs.

2 The Representation Language

Throughout this paper, we use P and sometimes P^* to denote a probability distribution; $A, B, \ldots$ with possible subscripts to denote atoms; names starting with a capital letter or X to denote domain variables; and $p, q, \ldots$ with possible subscripts to denote predicates. There are two types of predicates: *context and probabilistic predicates.*

Context predicates (c-predicates) have value true or false and are deterministic. They are used to describe the context the agent is in and to eliminate unnecessary or inappropriate probabilistic information from consideration by the agent. An atom formed from a context predicate is called a *context atom (c-atom).* A *context literal* is either a c-atom or the negation of a c-atom. A *context base* is a normal logic program [11], which is a set of universally quantified sentences of the form $C_0 \leftarrow L_1, L_2, ..., L_n$, where n is a nonnegative integer, $\leftarrow$ stands for implication, comma for logical conjunction, C_0 for a c-atom, and $L_i, 1 \leq i \leq n$, for context literals. We use *completed logic programs* proposed by Clark [11] for the semantics of the context base.

Each probabilistic predicate (p-predicate) represents a class of similar random variables. P-predicates appear in probabilistic sentences and are the focus of probabilistic inference processes. An atom formed from a probabilistic predicate is called a *probabilistic atom (p-atom).* Queries and evidence are expressed as p-atoms. In the probability models we consider, each random variable can assume a value from a finite set and in each possible realization of the world, that variable can have one and only one value. We capture this property by requiring that each p-predicate has at least one attribute. The last attribute of a p-predicate represents the *value* of the corresponding variable. For example, the variable *neighborhood* of a person x can have value *bad, average, good* and can be represented by a two-position predicate—the first position indicates the person and the second indicates the type of that person's neighborhood. Associated with each p-predicate p must be a statement of the form $VAL(p) = \{v_1, \ldots, v_n\}$, where $v_1, \ldots, v_n$ are constants denoting the possible values of the corresponding random variables. Let $A = p(t_1, \ldots, t_{m-1}, t_m)$ be a p-atom, we use $obj(A)$ to designate the tuple $(p, t_1, \ldots, t_{m-1})$ and $val(A)$ to designate t_m. If A is an atom of predicate p, then $VAL(A)$ means $VAL(p)$. If A is a ground p-atom then $obj(A)$ represents a concrete

random variable or object in the model and $val(A)$ is its value. We also define $Ext(A)$, *the extension of* A, to be the set $\{ p(t_1, \ldots, t_{m-1}, v_i) | 1 \leq i \leq n\}$. If A is a ground p-atom, we assume that the elements of $Ext(A)$ are mutually exclusive and exhaustive: in each "possible world", one and only one element in $Ext(A)$ is true. We assume that the arguments to p-predicates are typed, so that each argument is assigned values in some well-defined domain. We denote the set of all such predicate declarations in a knowledge base by *PD*.

Let A be a (probabilistic or context) atom. We define $ground(A)$ to be the set of all ground instances of A. A set of ground p-atoms $\{A_i | 1 \leq i \leq n\}$ is called *coherent* if there does not exist any A_j and $A_{j'}$ such that $j \neq j'$ and $obj(A_j) = obj(A_{j'})$ (and $val(A_j) \neq val(A_{j'})$).

A probabilistic sentence has the form $(P(A_0|A_1, \ldots, A_n) = \alpha) \leftarrow L_1, \ldots, L_m$ where $n \geq 0, m \geq 0, 0 \leq \alpha \leq 1, A_i$ are p-atoms, and L_j are context literals. The sentence can have free variables and each free variable is universally quantified over the entire scope of the sentence. The above sentence is called context-free if $m = 0$. If S is the above probabilistic sentence, we define $context(S)$ to be the conjunction $L_1 \wedge \ldots \wedge L_m$, $prob(S)$ to be the probabilistic statement $P(A_0|A_1, \ldots, A_n) = \alpha$, $ante(S)$ to be the conjunction $A_1 \wedge \ldots \wedge A_n$ and $cons(S)$ to be A_0. Sometimes, we use $ante(S)$ as the set of conjuncts and call A_0 the consequent of S. A probabilistic base (PB) of a knowledge base is a finite set of probabilistic sentences.

A PB will typically not be a complete specification of a probabiltiy distribution over the random variables represented by the p-atoms. One type of information which may be lacking is the specification of the probability of a variable given combinations of values of two or more variables which influence it. For real-world applications, this type of information can be difficult to obtain. For example, for two diseases D_1 and D_2 and a symptom S we may know $P(S|D_1)$ and $P(S|D_2)$ but not $P(S|D_1, D_2)$. Combining rules such as generalized noisy-OR [5, 16] are commonly used to construct such combined influences.

We define *a combining rule* as any algorithm that takes as input a set of *ground* context-free probabilistic sentences with the same consequent
$\{P(A_0|A_{i1}, \ldots, A_{in_i}) = \alpha_i | 1 \leq i \leq m\}$ such that $\cup_{i=1}^{m} \{A_{i1}, \ldots, A_{in_i}\}$ is coherent and produces as output $P(A_0|A_1, \ldots, A_n) = \alpha$, where A_1,..., and A_n are all different and $\{A_1, \ldots, A_n\}$ is a subset of $\cup_{i=1}^{m} \{A_{i1}, \ldots, A_{in_i}\}$. We assume that for each p-predicate p, there exists a corresponding combining rule in CR, the combining rules component of the KB.

A knowledge base (KB) consists of predicate descriptions, a probabilistic base, a context base, and a set of combining rules. We write $KB = \langle PD, PB, CB, CR \rangle$. Figure 2 shows a possible knowledge base for representing the burglary example introduced in the previous section.

We have the following p-predicates: *nbrhd*, *burglary*, *quake*, *alarm*, and *tornado*. The statements $nbrhd(Somebody : Person, V)$ and $VAL(nbrhd) = \{bad, average, good\}$ say that the first argument of *nbrhd* is a value in the domain *Person*, which is the set of all persons and that V can take one value in the set $\{bad, average, good\}$. For a person, say named John, $nbrhd(john, good)$ means the random variable *neighborhood of John*, indicated in the language by $obj(nbrhd(john, good))$, is *good*, indicated in the language by

$$
\begin{aligned}
PD &= \{ \quad nbrhd(Somebody : Person, V), VAL(nbrhd) = \{bad, average, good\}; \\
&\quad\; burglary(Somebody : Person, V), VAL(burglary) = \{yes, no\}; \\
&\quad\; quake(Somearea : Area, V), VAL(quake) = \{yes, no\}; \\
&\quad\; alarm(Somebody : Person, V), VAL(alarm) = \{yes, no\}; \\
&\quad\; tornado(Somearea : Area, V), VAL(tornado) = \{yes, no\}\} \\
PB &= \{ \quad P(nbrhd(X, average)) = .4 \\
&\quad\; P(nbrhd(X, bad)) = .3 \leftarrow in\_CALI(X) \\
&\quad\; P(nbrhd(X, good)) = .3 \leftarrow in\_CALI(X) \\
&\quad\; P(nbrhd(X, bad)) = .2 \leftarrow in\_WISC(X) \\
&\quad\; P(nbrhd(X, good)) = .4 \leftarrow in\_WISC(X) \\
&\quad\; P(burglary(X, yes) | nbrhd(X, average)) = .4 \\
&\quad\; P(burglary(X, yes) | nbrhd(X, good)) = .3 \leftarrow in\_CALI(X) \\
&\quad\; P(burglary(X, yes) | nbrhd(X, bad)) = .6 \leftarrow in\_CALI(X) \\
&\quad\; P(burglary(X, yes) | nbrhd(X, good)) = .2 \leftarrow in\_WISC(X), burglarized(X) \\
&\quad\; P(burglary(X, yes) | nbrhd(X, bad)) = .4 \leftarrow in\_WISC(X), burglarized(X) \\
&\quad\; P(burglary(X, yes) | nbrhd(X, good)) = .15 \leftarrow in\_WISC(X), \neg burglarized(X) \\
&\quad\; P(burglary(X, yes) | nbrhd(X, bad)) = .3 \leftarrow in\_WISC(X), \neg burglarized(X) \\
&\quad\; P(alarm(X, yes) | tornado(Y, yes)) = .99 \leftarrow in\_WISC(X), district(X, Y) \\
&\quad\; P(alarm(X, yes) | tornado(Y, no)) = .1 \leftarrow in\_WISC(X), district(X, Y) \\
&\quad\; P(alarm(X, yes) | burglary(X, yes)) = .98 \\
&\quad\; P(alarm(X, yes) | burglary(X, no)) = .05 \\
&\quad\; P(alarm(X, yes) | quake(Y, yes)) = .99 \leftarrow in\_CALI(X), district(X, Y) \\
&\quad\; P(alarm(X, yes) | quake(Y, no)) = .15 \leftarrow in\_CALI(X), district(X, Y) \\
&\quad\; \cdots\} \\
CB &= \{ \quad in\_CALI(X) \leftarrow \neg in\_WISC(X) \\
&\quad\; live\_in(X, Z) \leftarrow live\_in(X, Y), in(Y, Z) \\
&\quad\; live\_in(X, Y) \leftarrow district(X, Y) \\
&\quad\; in\_WISC(X) \leftarrow live\_in(X, wisconsin) \\
&\quad\; in\_CALI(X) \leftarrow live\_in(X, california) \\
&\quad\; in(madison, wisconsin)\} \\
\\
CR &= \{ \quad Generalized - Noisy - OR\}
\end{aligned}
$$

Figure 2: Example knowledge base.

$val(nbrhd(john, good)) = good$. Similar interpretations apply to other statements in PD.

PB contains the probabilistic sentences. Due to space limitations, we cannot show all the sentences. We would have sentences describing the probabilities the variables corresponding to *burglary* and *alarm* achieving the value *no*, and the marginal probabilities for *quake* and *tornado*. CB specifies relationships in the context information. For example, the first clause, $in\_CALI(X) \leftarrow \neg in\_WISC(X)$, states our (simplified) closed world assumption that if a person does not live in Wisconsin then he lives in California and due to the completed semantics, the reverse is also true. The negation in the antecedent encodes a non-monotonic deduction by negation as failure. Finally, Generalized Noisy-OR is used as the combining rule for all p-predicates.

3 Declarative Semantics

3.1 The Relevant Knowledge Base

It can be difficult for a user to guarantee global consistency of a large probabilisitic knowledge base, especially when we allow context-dependent specifications. We define our semantics so that we need consider only that portion of a knowledge base relevant to a given problem. Thus if part of the knowledge base is inconsistent, this will not affect reasoning in all contexts. We define the semantics relative to an inference session, characterized by a set of evidence and a set of context information.

Definition 1 *A set of* **evidence** E *is simply a set of p-atoms such that* $ground(E)$ *is coherent. A set of* **context information** C *is any set of c-atoms.*

In a particular inference session, characterized by a set of context information and a set of evidence, only a portion of the KB will be relevant. The set of relevant atoms is the set of evidence, the set of atoms whose marginal probability is directly stated, and the set of atoms influenced by these two sets, as indexed by the context information. In constructing the set of relevant p-atoms, we consider only the qualitative dependencies described by probabilistic sentences. If $(P(A_0|A_1, \ldots, A_n) = \alpha) \leftarrow L_1, \ldots, L_m$ is a ground instance of a sentence in PB conforming to type constraints and $L_1 \wedge \ldots \wedge L_m$ can be deduced from $completed(C \cup CB)$, then that sentence confirms the fact that A_0 *is directly influenced (or caused) by* $A_1, \ldots, A_n$ in the current context. If, in addition, $A_1, \ldots, A_n$ are relevant atoms then it is natural to consider A_0 as relevant. Let $completed(C \cup CB)$ be the completed logic program with the associated equality theory [11] constructed from $C \cup CB$, then we have the following definition of the set of relevant p-atoms.

Definition 2 *Given a set of evidence* E*, a set of context information* C *and a KB, the set of* **relevant p-atoms (RAS)** *is defined recursively by:*

(1) $ground(E) \subseteq RAS$*;*

(2) if S *is a ground instance of a probability sentence, conforming to type constraints, such that* $context(S)$ *is a logical consequence of* $completed(C \cup CB)$ *and* $ante(S) \subseteq RAS$ *then* $cons(S) \in RAS$*;*

(3) if a p-atom A *is in* RAS *then* $Ext(A) \subseteq RAS$*;*

(4) RAS is the smallest set statisfying the above conditions.

The RAS is constructed in a way similar to Herbrand least models for Horn programs. Context information is used to eliminate the portion of PB which is not related to the current problem.

Proposition 1 *Given a set of evidence E, a set of context information C and a KB, RAS always exists.*

Proof The proof is similar to that in [11] for the existence of the least Herbrand model for a Horn program. □

Definition 3 *Given a set of evidence E, a set of context information C and a KB, the set of* **relevant probabilistic sentences (RPB)** *is defined as the set of all prob(S), such that S is a ground probabilistic sentence, context(S) is a logical consequence of completed($C \cup CB$), $cons(S) \in RAS$, and $ante(S) \subseteq RAS$.*

The RPB contains the basic causal relationships between p-atoms in RAS. In the case of multiple causes represented by multiple sentences, we need combining rules to construct the combined probabilistic influence.

Definition 4 *Given a set of evidence E, a set of context information C, and a KB, the combined RPB (CRPB) is constructed by applying the applicable combining rule to each maximal set of sentences $\{S_1, \ldots, S_n\}$ in RPB which have the same consequent and such that $\cup_{i=1}^{n} ante(S_i)$ is coherent.*

Combined RPB's play a similar role to completed logic programs. We assume that each sentence in CRPB describes all random variables which directly cause the random variable in the consequent. Given a set of evidence E, a set of context information C and a KB, we can construct CRPB, as the following example shows.

Example 1 *Consider our example KB and suppose we have the evidence E={burglary(john, yes)} and the context C={district(john,madison), burglarized(john)}. In this case,*
RAS={ nbrhd(john,bad), nbrhd(john,average), nbrhd(john,good), burglary(john,yes), burglary(john,no), alarm(john,true), alarm(john,false), tornado(madison,yes), tornado(madison,no)}, and RPB = {P(nbrhd(john,average))=.4, P(nbrhd(john,bad))=.2, P(nbrhd(john,good))=.4, P(burglary(john,yes)|nbrhd(john,average))=.4,
P(burglary(john,yes)|nbrhd(john,good))=.2,
P(burglary(john,yes)|nbrhd(john,bad))=.4,
P(alarm(john,yes)|tornado(madison,yes))=.99,
P(alarm(john,yes)|burglary(john,yes))=.98,
P(alarm(john,yes)|burglary(john,no))=.05,
P(alarm(john,yes)|tornado(madison,no))=.1, $\cdots$ }.
In the CRPB, the sentences in RPB with alarm as the consequent are transformed into sentences specifying the probability of alarm conditioned on both burglary and tornado. The other sentences in RPB remain the same in CRPB.

We define a syntactic property of CRPB which is necessary in constructing Bayesian networks from the KB.

Definition 5 *A CRPB is* **completely quantified** *if*

(1) for all ground atoms A_0 in RAS, there exists at least one sentence in CRPB with A_0

in the consequent; and

(2) for all ground sentences S in CRPB we have the following property: Let S have the form $P(A_0|A_1,...,A_n) = \alpha$*, then for all* $i = 0,..,n$*, if* $val(A_i) = v$ *and* $v' \in VAL(A_i), v \neq v'$*, there exists another ground sentence S' in CRPB such that S' can be constructed from S by replacing* $val(A_i)$ *by* v' *and* α *by some* α'*.*

If we think of each ground $obj(A)$, where A is some p-atom, as representing a random variable in a Bayesian network model then the above condition implies that we can construct a link matrix for each random variable in the model. This is a generalization of constraint (C1) in [9]. We do not require the existence of link matrix for every random variable, but only for the random variables that are relevant to an inference problem.

3.2 Probabilistic Independence Assumption

Beside the probabilitistic quantities given in a PB, we assume some probabilistic independence relationships specified by the structure of probabilistic sentences. Probabilistic independence assumptions are used in all related work [17, 3, 8, 15, 9] as the main device to construct a probability distribution from local conditional probabilities. Unlike Poole [15] who assumes independence on the set of consistent *"assumable"* atoms, we formulate the independence assumption in our framework by using the structure of the sentences in CRPB. We find this approach more natural since the structure of the CRPB tends to reflect the causal structure of the domain and independencies are naturally thought of causally.

Definition 6 *Given a set of ground context-free probabilistic sentences, let A and B be two ground p-atoms. We say A is* **influenced by** *B if*

(1) there exists a sentence S, an atom A' *in* $Ext(A)$ *and an atom* B' *in* $Ext(B)$ *such that* $A' = cons(S)$ *and* $B' \in ante(S)$ *or*

(2) there exists another ground p-atom C such that A is influenced by C and C is influenced by B.

Example 2 *Continuing the burglary example,* $alarm(john, yes)$ *is influenced by* $nbrhd(john, good)$, $nbrhd(john, bad)$, $burglary(john, yes)$ *and* $tornado(madison, no)$*.*

Assumption *We assume that if* $P(A_0|A_1,...,A_n) = \alpha$ *is in CRPB then for all ground p-atoms B which are not in* $Ext(A_0)$ *and not* influenced by A_0, A_0 *and B are probabilistically independent given* $A_1,...,A_n$*.*

Example 3 *Continuing the burglary example, alarm(john,yes) is probabilistically independent of nbrhd(john,good) and nbrhd(john,bad) given burglary(john,yes) and tornado(madison,no).*

3.3 Model Theory

We define the semantics by using possible worlds on the Herbrand base. This approach of characterizing the semantics by canonical Herbrand models is widely used in work on logic programming [6, 7]. In our semantics, the context constraints in the context base CB and the set of context information C act to select the appropriate set of the possible worlds over

which to evaluate the probabilistic part of the sentences. In this way they index probability distributions.

The RAS contains all relevant atoms for an inference session. We assume that *in such a concrete situation, the belief of an agent can be formulated in terms of possible models on RAS.*

Definition 7 *Given a set of evidence E, a set of context information C and a KB, a* **possible model** *M of the corresponding CRPB is a set of ground atoms in RAS such that for all A in RAS, $Ext(A) \cap M$ has one and only one element. The atoms in M represent the set of atoms assigned the value true; all other atoms are assigned the value false.*

A probability distribution on the possible models is realized by probability density assignment to each model. Let P be a probability distribution on the possible models, we define $P(A_1, \ldots, A_n)$, where $A_1, \ldots, A_n$ are atoms in RAS, as $\sum\{P(w)|w$ is a possible model containing $A_1, \ldots, A_n\}$. We take a sentence of the form $P(A_0|A_1, \ldots, A_n) = \alpha$ as shorthand for $P(A_0, A_1, \ldots, A_n) = \alpha \times P(A_1, \ldots, A_n)$, so that probabilities conditioned on zero are not problematic. We say P satisfies a sentence $P(A_0|A_1, \ldots, A_n) = \alpha$ if $P(A_0, A_1, \ldots, A_n) = \alpha \times P(A_1, \ldots, A_n)$ and P satisfies CRPB if it satisfies every sentence in CRPB.

Definition 8 *A probability distribution* **induced by** *the set of evidence E, the set of context information C, and KB is a probability distribution on possible models of CRPB satisfying CRPB and the independence assumptions implied by CRPB.*

Example 4 *For an incompletely quantified CRPB, there are many possible distributions. For example, if $CRPB = \{P(nbrhd(john, bad)) = .5, P(tornado(madison, yes)) = .6,$ $P(tornado(madison, no)) = .4\}$ and nbrhd can take one of three values $bad, average$ and good, then any probability assignment P^* such that $P^*(nbrhd(john, bad)) = .5$ and $P^*(nbrhd(john, average)) + P^*(nbrhd(john, good)) = .5$ is an induced probability distribution.*

We define the consistency property only on the relevant part of a KB. Since the entire KB contains information about various contexts, testing for consistency of such a KB may be very difficult.

Definition 9 *A completely quantified CRPB is* **consistent** *if*
(1) there is no atom in RAS which is influenced by itself and
(2) for all $P(A_0|A_1, ..., A_n) = \alpha$ in CRPB, $\sum\{\alpha_i|P(A_0'|A_1, ..., A_n) = \alpha_i \in CRPB$ and $obj(A_0') = obj(A_0)\} = 1$.

Condition (1) rules out cycles and condition (2) enforces the usual probability assignment constraint.

The following example shows that PB may contain loops but a specific completely quantified CRPB is still consistent.

Example 5 *Let $PB = \{P(p(true)|p(true)) = 1; P(p(true)|p(false)) = 1;$ $P(p(false)|p(true)) = 1; P(p(false)|p(false)) = 1; P(q(true)|r(true)) = .7;$ $P(q(false)|r(true)) = .3; P(q(true)|r(false)) = .5; P(q(false)|r(false)) = .5\}$. PB contains a causal loop and the probability assignment is not reasonable. Assume the set of evidence is $\{r(true)\}$. The CRPB is $\{P(q(true)|r(true)) = .7; P(q(false)|r(true)) = .3;$*

$P(q(true)|r(false)) = .5; P(q(false)|r(false)) = .5\}$ *is completely quantified and consistent.*

Theorem 1 *Given a set of evidence E, a set of context information C and a KB, if RAS is finite and the completely quantified CRPB is consistent then there exists one and only one induced probability distribution.*

Proof Let M be a possible model of CRPB. We arrange the atoms in M in the order of the influenced-by relationship: If A_i is influenced by A_j in CRPB then $j > i$. Let P^* be a probability distribution induced by CRPB. We have $P^*(M) = P^*(\wedge_{i=1} A_i) = P^*(A_1 | \wedge_{i=2} A_i) \times P^*(\wedge_{i=2} A_i) = P(A_1 | A_{11}, \ldots, A_{1n}) \times P^*(\wedge_{i=2} A_i) = \alpha \times P^*(\wedge_{i=2} A_i)$, where $P(A_1 | A_{11}, \ldots, A_{1n}) = \alpha$ is a sentence in CRPB with $\{A_{11}, \ldots, A_{1n}\} \subseteq M$, by the independence assumption. By using induction on the length of the sequence, we get a unique P^*. □

3.4 Probabilistic Queries

In one inference session, we can pose queries to ask for the posterior probabilities of some random variables.

Definition 10 *A* **complete ground query** *wrt the set of evidence E and the set of context information C is a query of the form $P(Q) =?$, where the last argument of Q is a variable and it is the only variable in Q. The meaning of such a query is: find the posterior probability distribution of $obj(Q)$. If $VAL(Q) = \{v_1, ..., v_n\}$ then the answer to such a query is a vector $\vec{\alpha} = (\alpha_1, ..., \alpha_n)$, where $0 \leq \alpha_i \leq 1, \sum_{i=1}^{n} \alpha_i = 1$ and α_i is the posterior probability of $obj(Q)$ receiving the value v_i.*

A **complete query** *is a query of the form $P(Q) =?$, where the last argument of Q is a variable and the other arguments may also contain variables.*

Example 6 *We can pose the following complete ground query to the example KB: $P(alarm(john, V)) =?$ to ask for the posterior probability of John's alarm status.*

An inference problem in an inference session involving a KB, a set of evidence E, a set of context information C is characterized by a query.

Definition 11 *Given a set of evidence E, a set of context information C, a KB and a complete ground query $P(Q) =?$. Assume that $VAL(Q) = \{v_1, ..., v_n\}$. We say $P(Q) = (\alpha_1, ..., \alpha_n)$ is a* **logical consequence** *of $\langle E, C, KB \rangle$ if for all probability distributions P^* induced by E, C and KB, $\forall 0 \leq i \leq n : P^*(\{M | Q_i$ and E are true in $M\}) = \alpha_i \times P^*(\{M | E$ are true in $M\})$, where Q_i is Q after replacing $val(Q)$ by v_i.*

Example 7 *Continuing example 4, suppose we have the query $Q = tornado(madison, V)$. There are an infinite number of induced probability distributions but in all of them $P(tornado(madison, yes)) = .6$ and $P(tornado(madison, no)) = .4$. So they are logical consequences of CRPB.*

In order to prove that the answers returned by our query answering procedure are correct, we need the following definition.

Definition 12 *Given a set of evidence E, a set of context information C, a KB and a complete ground query $P(Q) =?, P(Q) = (\alpha_1, ..., \alpha_n)$ is a* **correct answer** *to the complete*

ground query $P(Q) =?$ *if* $P(Q) = (\alpha_1, ..., \alpha_n)$ *is a logical consequence of* $\langle E, C, KB \rangle$.

If $P(Q) =?$ *is a complete query then* $P(Q') = \alpha$ *is a* correct answer *to the complete query* $P(Q) =?$ *if* Q' *is a ground instance of* Q *and* $P(Q') = \alpha$ *is a correct answer to the complete ground query* $P(Q') =?$.

Definition 13 *Given a set of evidence E, a set of context information C and a KB. We say a query answering procedure is* **sound** *(wrt complete queries) if for any complete query* $P(Q) =?$, *any answer returned by the procedure is correct. We say a query answering procedure is* **complete** *(wrt complete queries) if for any complete query* $P(Q) =?$, *any correct answer is returned by the procedure after a finite amount of time.*

4 Query Answering Procedure

In this section we present an algorithm, Q-prodecure, for answering complete queries. We assume that the *domains are finite*[1]. Assume that we are given $P(Q) =?, E, C$, and KB, where $P(Q) =?$ is a complete query. In order to answer contextual queries on CB, i.e., to determine whether a probabilistic sentence is in the current context, Q-procedure frequently calls the SLDNF [11] proof procedure which works on $C \cup CB$ and queries provided by Q-procedure. SLDNF is sound and, for some classes of normal logic programs, complete under completed program semantics. Q-procedure is more complex than SLDNF partially because it needs to collect all relevant sentences before combining rules can be used. Q-procedure also calls a Bayesian Network belief updating procedure. Several such procedures are available [14].

Q-procedure has the following steps: build the necessary portion of the Bayesian network, each node of which corresponds to an $obj(A)$, where A is a ground p-atom in RAS; update the Bayesian network using the set of evidence E; and output the updated belief of the query nodes. The main idea of the algorithm is to build a *supporting network* for each random variable corresponding to a ground instance of an evidence atom or the query.

Definition 14 *Let A be a ground p-atom and consider the set of all ground p-atoms B such that A is influenced by B in CRPB. The* **supporting network** *for* $obj(A)$ *is a Bayesian network consisting of* $obj(A)$ *and the set of all* $obj(B)$, *with the relevant "influenced by" relationships represented as links or sequences of links. Let* $A_1, \ldots, A_n$ *be ground p-atoms. The supporting network for* $obj(A_1), \ldots, obj(A_n)$ *is the union of the supporting networks for each* $obj(A_i), 1 \leq i \leq n$.

Example 9 *The supporting network of* $obj(burglary(john, yes))$ *consists of two nodes:* $obj(burglary(john, .))$ *and* $obj(nbrhd(john, .))$ *with the corresponding link matrices.*

Constructing the network by building supporting networks is justified by the fact that atoms which do not influence either the evidence or the query are irrelevant. To build the supporting networks for the evidence, we first generate the set of all ground instances of the evidence p-atoms. Then for each ground instance, we build the supporting network

[1]Q-procedure is designed to present the technique behind network construction as straightforwardly as possible by sacrificing efficiency. Efficient network construction will be the subject of a future paper.

using PB, the set of ground instances of the evidence which have not been explored, and the current net. In building the supporting network for an evidence atom, we need to consider other evidence because in defining RAS we use the entire set of evidence.

The pseudo code for Q-procedure is shown in Figure 3. The FOR loop constructs the supporting network for the set of evidence and the final BUILD-NET call augments the constructed network with supporting network of ground instances of the query. UPDATE(NET,E) is any probability propogation algorithm on Bayesian networks.

The supporting networks are constructed via calls to the BUILD-NET function. BUILD-NET receives as input an atom, whose supporting network needs to be explored. It updates the NET, which might have been partially built. The return value of the function is a set of substitutions such that for each substitution there exists a supporting network for the ground instance of the atom corresponding to the substitution. BUILD-NET frequently calls SLDNF to answer queries on CB $\cup$ C.

In BUILD-NET, the RS variable is used to collect all sentences in RPB which have the input atom as consequent. The COMBINE function uses the appropriate combining rule in CR and the sentences in RS to generate the corresponding sentences in the CRPB. The variable SUBSS stores all substitutions which produce ground instances of the input atom A corresponding to random variables that appear in the constructed network. The recursive structure of BUILD-NET is realized through the function BUILD-CONJUNCTION-NET which builds the supporting network for the atoms in the antecedents of relevant sentences.

BUILD-CONJUNCTION-NET receives as input a conjunction of atoms, whose supporting networks need to be explored. It updates the NET, which might have been partially built. The return value of the function is a set of substitutions such that for each substitution all the ground instances resulting from applying the substitution to the atoms in the conjuction appear in the resulting net.

4.1 The correctness of Q-procedure

In Q-procedure, we construct only the supporting networks, not the entire network for CRPB. The following theorem shows that this portion of the network is sufficient for evaluating the given query.

Theorem 2 *Given a complete query $P(Q) =?$, a set of evidence E, a set of context information C and a KB. If the domains are finite, CRPB is completely quantified and consistent, the proof procedure for $C \cup CB$ is sound and complete wrt any context query generated by Q-procedure and Q-procedure stops after a finite amount of time then:*

(1) there exists a Bayesian network model of the unique probability distribution induced by CRPB;

(2) there is a node in the constructed network for a ground instance of the query if and only if there is at least one sentence in CRPB with that instance as consequent;

(3) the network constructed by Q-procedure is the supporting network of random variables which correspond either to ground instances of the query or to ground instances of evidence;

(4) and the posterior probability distribution of any node in the constructed network does

Q-PROCEDURE

BEGIN
{ Build the subnetworks that supports the evidence }
NET:= {};
E_list := list of ground instances of $E_i \in E$ in any order of i;
FOR i:=1 TO $length\_of\_E\_list$ DO BEGIN
$E\_set := \cup_{j>i}(\{P(E\_list_j) = 1\} \cup \{P(E_k) = 0 | E_k$ is in $Ext(E\_list_j)$
but different from $E\_list_j\}$;
temp := BUILD-NET(E_list_i, NET, $PB \cup E\_set$);
IF temp={} THEN construct a node $obj(E\_list_i)$
with probability assignment $P(E\_list_i) = 1$
END;
{ Build the subnetworks that support the ground instances of the query }
SUBSS := BUILD-NET (Q, NET, PB);
{ Prune the isolated portions of the network }
FOR i:=1 TO $length\_of\_E\_list$ DO
IF there is no path connecting $obj(E\_list_i)$
and a node corresponding to a ground instance of query
THEN delete the supporting network of $obj(E\_list_i)$
{ Perform the probability propagation on NET}
UPDATE(NET,E);
{ Output posterior probabilities }
FOR each θ in SUBSS output the probability values at node $obj(Q\theta)$;
END.

Figure 3: Query processing procedure.

BUILD-NET

```
Function BUILD-NET( A: a p-atom; var NET: a Bayesian net; Extended_PB: a set of
     probabilistic sentences): set of substitutions;
var SUBSS, SUBSS1: a set of substitutions;
    RS, CRS: a set of probabilistic sentences;
    S: a probabilistic sentence;
    θ^S, θ^C, θ, θ': substitution;
BEGIN
    A:= A after replacing val(A) by a new variable name;
    SUBSS := {};
    FOR each θ such that Aθ is ground DO
     IF there is a node in NET corresponding to obj(Aθ)
        and the link matrix entry for val(Aθ) exists
     THEN  SUBSS := SUBSS ∪ {θ}
     ELSE
        BEGIN
            RS := {};
            FOR each S ∈ Extended_PB such that there exists
                an mgu θ^S of Aθ and cons(S) DO
                { Assume that if context(S) is empty then there is exactly
                    one θ^C, which is the empty substitution }
                FOR each θ^C which is a computed answer from
                    (← context(S)θ^S, C ∪ CB) returned by SLDNF DO
                    IF ante(S) = {}
                    THEN RS := RS ∪{prob(S)θ^S θ^C}
                    ELSE  BEGIN
                              SUBSS1:= BUILD-CONJUNCTION-NET(
                                           ante(S)θ^S θ^C, NET,Extended_PB);
                              RS:= RS ∪{prob(S)θ^S θ^C θ'|θ' ∈ SUBSS1}
                          END;
            CRS := COMBINE(RS, CR);
            IF CRS is not empty THEN
               BEGIN
                    SUBSS := SUBSS ∪ {θ};
                    IF obj(Aθ) is not in NET THEN
                       BEGIN construct node obj(Aθ);
                                build the links from the parents
                       END;
                    assign the corresponding link matrix entries;
               END
        END;
    RETURN SUBSS;
END.
```

Figure 4: BUILD-NET function.

BUILD-CONJUNCTION-NET

Function BUILD-CONJUNCTION-NET($A_1 \wedge ... \wedge A_n$: a conjunction of p-atoms; var NET:
a Bayesian net; $Extended\_PB$: a set of probabilistic sentences): set of substitutions;
var SUBSS: set of substitutions;
BEGIN
SUBSS := BUILD-NET (A_1, NET, $Extended\_PB$);
IF $n > 1$ THEN
FOR each $\theta \in$ SUBSS DO
SUBSS:= (SUBSS-$\{\theta\}$) $\cup\{\theta\theta' |\ \theta' \in$ BUILD-CONJUNCTION-NET(
$(A_2 \wedge \ldots \wedge A_n)\theta, NET$,Extended_PB)$\}$;
RETURN SUBSS;
END.

Figure 5: BUILD-CONJUNCTION-NET function.

not depend on any node not in the constructed network.

Proof (Sketch) We prove them in sequence.

(1) Simply apply Corrolary 4, Chapter 3 of [14]. Here each node of the network corresponds to an $obj(A)$, A is an atom in RAS; and there is a link from node $obj(B)$ to node $obj(A)$ if there is a probabilistic sentence $P(A|\ldots, B, \ldots) = \alpha$ in CRPB.

(2) Obvious.

(3) Proof by induction on the length of recursive calls of BUILD-NET.

(4) Follow the line of the theorem 1. □

Theorem 3 *(Soundness and completeness) Given a complete query $P(Q) =?$, a set of evidence E, a set of context information C and a KB, if the domains are finite, CRPB is completely quantified and consistent, the proof procedure for $C \cup CB$ is sound and complete wrt any context query generated by Q-procedure and Q-procedure stops after a finite amount of time then Q-procedure is sound and complete.*

Proof (Sketch) The proof is based on the previous theorem and the soundness of the Bayesian network updating procedure [14].□

4.2 The completeness of Q-procedure for Acyclic KBs

In this section, we show that Q-procedure is complete for a large class of KB, which is characterized by acyclicity syntactic rule.

Definition 15 *An CB is called acyclic if there is a mapping $C\_ord()$ from the set of ground instances of atoms in CB into the set of natural numbers such that for any ground instance $C_0 \leftarrow L_1, ..., L_n$ of some clause in CB, $C\_ord(C_0) > C\_ord(L_i), \forall i : 1 \leq i \leq n$, where we extend $C\_ord()$ by $C\_ord(\neg A) = C\_ord(A)$, for all atom A.*

A PB is called **acyclic** *if there is a mapping $P\_ord()$ from the set of ground instances of p-atoms in PB into the set of natural numbers such that for any ground instance $(P(A_0 \leftarrow A_1, ..., A_n) = \alpha) \leftarrow L_1, .., L_m$ of some clause in PB, $P\_ord(A_0) > P\_ord(A_i), \forall i : 1 \leq i \leq n$.*

The expressiveness of acyclic logic programs is demonstrated in [1]. We hope that acyclic PBs also have equal importance. To the best of our knowledge, PBs with loops are considered problematic and all PB's considered in the literature are acyclic.

Theorem 4 *Given a complete query $P(Q) =?$, a set of evidence E, a set of context information C and a KB, if the domains are finite and KB is acyclic (that means both PB and CB are acyclic) and CRPB is completely quantified and consistent then Q-procedure is sound and complete.*

Proof SLDNF is sound and complete for acyclic normal programs with *finite domains wrt an arbitrary query* because floundering queries can be avoided by instantiating them to ground. See [1] for more details. It is obvious that Q-procedure always stops for acyclic programs with finite domains. The result follows directly from theorem 3. □

5 Discussion and Related Work

This paper has presented a method for answering queries from context-sensitive probabilistic knowledge bases. Context is used to facilitate specification of probabilistic knowledge and to reduce the complexity of inference by focusing attention only on the relevant portions of that knowledge. In addition to the applicability of the approach to the management of probabilistic data bases, the developed techniques also have significant potential for application in the area of temporal probabilistic reasoning and probabilistic planning. In a related paper [13] we present a temporal variant of our logic. We describe the application of this framework to modeling temporal probabilistic processes and apply the approach to a particular medical problem. In a second paper [12] we show how to represent probabilistic actions in the context-constrained temporal probability logic and formalize plan evaluation as deduction and plan generation as abduction in this logic.

This work extends the framework presented in [9] in several significant ways. That work did not address the problem of representing context-sensitive knowledge. In addition, it imposed a number of constraints (C1 – C4) on the language in order to guarantee that a set of ground instances of the rules in the knowledge base was isomorphic to a Bayesian network. In this work we drop these constraints, which increases the expressive power of the langugae but requires us to specify combining rules. Our probabilistic independence assumption is simpler than that of d-separation for a knowledge base used in [9].

Our representation language has some similarities to Breese's Alterid [3]. Breese also uses a form of predicates similar to our p-predicates. Breese mixes logic program clauses with probabilistic sentences. In contrast, we seperate logic program clauses from probabilistic sentences and use context predicates to select the relevant probabilistic sentences. In our framework, the pure logical dependencies between p-atoms are represented by 0 and 1 probability values. That could increase the reasoning time. This problem will be the topic of future research. Breese does not provide a semantics for the knowledge base. As a result, his paper cannot prove the correctness of his query answering procedure. Breese's procedure does both backward and forward chaining. Because Q-procedure only chains backwards, we can extend the procedure for some infinite domains.

Poole [15] expresses an intention similar to ours: "there has not been a mapping between logical specifications of knowledge and Bayesian network representations ..". He provides such a mapping using probabilistic Horn abduction theory, in which knowledge is represented by Horn clauses and the independence assumption of Bayesian networks is explicitly stated. His work is developed along a different track than ours, however, by concentrating on using the theory for abduction. Our approach has several advantages over Poole's. We do not impose as many constraints on our representation language as he does. Probabilistic dependencies are simply directly represented in our language, while in Poole's language they are indirectly specified through the use of special predicates in the rules. Our probabilistic independence assumption is more intuitively appealing since it reflects the causality of the domain. Our framework is representationally richer than Poole's by allowing context-dependent specification.

References

[1] K. R. Apt and M. Bezem. Acyclic programs. *New Generation Computing*, pages 335–363, Sept 1991.

[2] F. Bacchus. Using first-order probability logic for the construction of Bayesian networks. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pages 219–226, July 1993.

[3] J.S. Breese. Construction of belief and decision networks. *Computational Intelligence*, 8(4):624–647, 1992.

[4] G.F. Cooper. The computational complexity of probabilistic inference using bayesian belief networks. *Artificial Intelligence*, 42(2-3):393–405, 1990.

[5] F.J. Diez. Parameter adjustment in bayes networks. the generalized noisy or-gate. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*, pages 99–105, Washington, D.C., July 1993.

[6] A. V. Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *JACM*, pages 620–650, July 1991.

[7] M. Gelfond and V. Liftschitz. The stable model semantics for logic programming. In *Proceedings of the Fifth International Conference and Symposium on Logic Programming*, pages 1070–1080, 1988.

[8] R.P. Goldman and E. Charniak. A language for construction of belief networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(3):196–208, March 1993.

[9] P. Haddawy. Generating Bayesian networks from probability logic knowledge bases. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 262–269, Seattle, July 1994.

[10] D. Heckerman and M.P. Wellman. Bayesian networks. *Communications of the ACM*, 38(3):27–30, March 1995.

[11] J. W. Lloyd. *Foundation of Logic Programming*. Springer-Verlag, 1987. second edition.

[12] L. Ngo and P. Haddawy. Plan projection as deduction and plan generation as abduction in a context-sensitive temporal probability logic. 1995. Submitted to UAI95. (Available via www at http://www.cs.uwm.edu/faculty/haddawy).

[13] L. Ngo, P. Haddawy, and J. Helwig. A theoretical framework for context-sensitive temporal probability model construction with application to probabilistic plan projection. 1995. Submitted to UAI95. (Available via www at http://www.cs.uwm.edu/faculty/haddawy).

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

[15] D. Poole. Probabilistic horn abduction and bayesian networks. *Artificial Intelligence*, 64(1):81–129, November 1993.

[16] S. Srinivas. A generalization of the noisy-or model. In *UAI-93*, pages 208–217, July 1993.

[17] M.P. Wellman, J.S. Breese, and R.P. Goldman. From knowledge bases to decision models. *The Knowledge Engineering Review*, 7(1):35–53, 1992.