MEBN: A Language for First-Order Bayesian Knowledge Bases

Kathryn Blackmond Laskey KLASKEY@GMU.EDU
Department of Systems Engineering and Operations Research
MS4A6
George Mason University
Fairfax, VA 22030, USA

Abstract

Although classical first-order logic is the *de facto* standard logical foundation for artificial intelligence, the lack of a built-in, semantically grounded capability for reasoning under uncertainty renders it inadequate for many important classes of problems. Probability is the best-understood and most widely applied formalism for computational scientific reasoning under uncertainty. Increasingly expressive languages are emerging for which the fundamental logical basis is probability. This paper presents Multi-Entity Bayesian Networks (MEBN), a first-order language for specifying probabilistic knowledge bases as parameterized fragments of Bayesian networks. MEBN fragments (MFrags) can be instantiated and combined to form arbitrarily complex graphical probability models. An MFrag represents probabilistic relationships among a conceptually meaningful group of uncertain hypotheses. Thus, MEBN facilitates representation of knowledge at a natural level of granularity. The semantics of MEBN assigns a probability distribution over interpretations of an associated classical first-order theory on a finite or countably infinite domain. Bayesian inference provides both a proof theory for combining prior knowledge with observations, and a learning theory for refining a representation as evidence accrues. A proof is given that MEBN can represent a probability distribution on interpretations of any finitely axiomatizable first-order theory.

Keywords: Bayesian network, graphical probability models, knowledge representation, multi-entity Bayesian network, probabilistic logic, uncertainty in artificial intelligence

1 Introduction

First-order logic is primary among logical systems from both a theoretical and a practical standpoint. It has been proposed as a unifying logical foundation for defining extended logics and interchanging knowledge among applications written in different languages. However, its applicability has been limited by the lack of a coherent semantics for plausible reasoning. Among the many proposed logics for plausible inference, probability is the strongest contender as a universal standard of comparison for plausible reasoning systems. Probability has proved its worth in applications from a wide variety of problem domains, and is a rationally justified calculus for plausible inference under uncertainty (e.g., de Finetti, 1934/1975; Howson and Urbach, 1993, Jaynes, 2003; Savage, 1954).

Application of probability to complex, open-world problems requires languages based on expressive probabilistic logics. The development of sufficiently expressive probabilistic logics has been hindered by the lack of modularity of probabilistic reasoning, the intractability of worst-case probabilistic inference, and the difficulty of ensuring that probability assessments give rise to a well-defined and unique probability distribution. The number of probabilities required to express a fully general probability distribution over truth-values of a collection of assertions is exponential in the number of assertions, making a brute-force approach to specification and

inference infeasible for all but the simplest problems. These difficulties have been addressed by exploiting independence relationships to achieve parsimonious representation and efficient inference (Pearl, 1988; Neapolitan, 2003). Recent years have seen a rapid evolution of increasingly powerful languages for computational probabilistic reasoning (e.g., Buntine, 1994; D'Ambrosio, et al, 2001; Getoor et al, 2001; Gilks et al, 1994; Glesner and Koller, 1995; Heckerman, et al., 2004; Jaeger, 2001; Kersting and De Raedt, 2001; Koller and Pfeffer, 1997; Laskey and Costa, 2005; Laskey and Mahoney, 1997; Milch, et al., 2005; Ngo and Haddawy, 1997; Poole, 2003; Pfeffer, 2001; Sato, 1998; Spiegelhalter, et al., 1996).

This paper presents multi-entity Bayesian networks (MEBN), a language for representing first-order probabilistic knowledge bases. The fundamental unit of representation in MEBN is the MFrag, a parameterized Bayesian network fragment that represents uncertain relationships among a small collection of related hypotheses. MFrags allow knowledge to be specified at a natural level of granularity. Dependence relationships and local distributions are specified for conceptually meaningful clusters of related hypotheses. An MFrag can be instantiated multiple times by binding its arguments to different entities. MEBN thus provides a compact language for expressing complex graphical models with repeated structure. A MEBN theory consists of a set of MFrags that satisfies consistency conditions ensuring existence of a unique probability distribution over its random variables. MEBN theories can be used to reason consistently about complex expressions involving nested function application, arbitrary logical formulas, and quantification.

The remainder of the paper is organized as follows. Section 2 provides an overview of formalisms for knowledge representation and reasoning under uncertainty. Section 3 defines the MEBN language. Section 4 defines semantics, presents results on expressive power, and discusses inference. Section 5 reviews current research on expressive first-order languages. The final section is a summary and discussion. Proofs and algorithms are given in the appendix.

2 Probability and Logic

Davis (1990) defines a logic as a schema for defining languages to describe and reason about entities in different domains of application. Certain key issues in representation and inference arise across a variety of application domains. A logic encodes particular approaches to these issues in a form that can be reused across languages, domains, and theories.

By far the most commonly used, studied, and implemented logical system is first-order logic (FOL), invented independently by Frege and Peirce in the late nineteenth century (Frege, 1879/1967; Peirce, 1885). First-order logic is applied by defining a set of axioms, or sentences that make assertions about a domain. The axioms, together with the set of logical consequences of the axioms, comprise a theory of the domain. Until referents for the symbols are specified, a theory is a syntactic structure devoid of meaning. An interpretation for a theory specifies a definition of each constant, predicate and function symbol in terms of the domain. Each constant symbol denotes a specific entity; each predicate denotes a set containing the entities for which the predicate holds; and each function symbol denotes a function defined on the domain. The logical consequences of a set of axioms consist of the sentences that are true in all interpretations, also called the valid sentences.

Special-purpose logics built on first-order logic give pre-defined meaning to reserved constant, function and/or predicate symbols. Such logics provide built-in constructs useful in applications. There are logics that provide constants, predicates, and functions for reasoning about types, space and time, parts and wholes, actions and plans, etc. When a logic is applied to

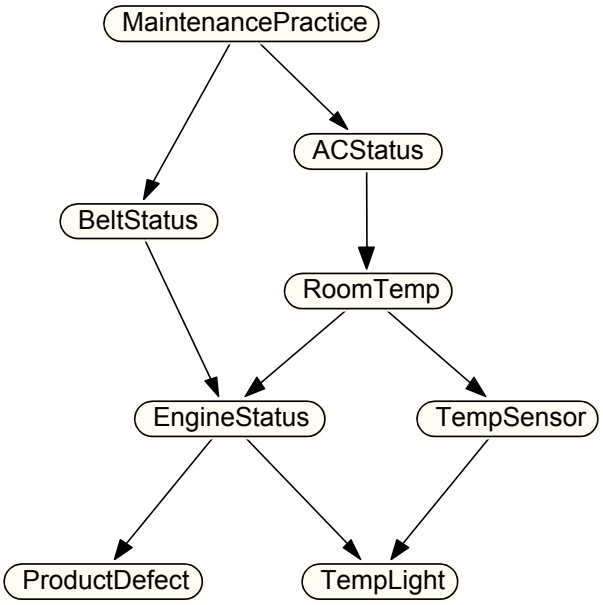


Figure 1: Bayesian Network for Diagnostic Task

reason about a particular domain, the modeler assigns meaning to additional domain-specific symbols, and provides axioms to assert important properties of their intended referents. Formal ontologies (Gruber, 1993; Sowa, 2000) are usually expressed in languages based on first-order logic or one of its subsets.

A first-order theory implies truth-values for the valid sentences and their negations, but provides no means to evaluate the plausibility of other sentences. Plausible reasoning is fundamental to intelligence, and plausible reasoning logics have been an active area of research in artificial intelligence. Because probability is not truth-functional, naïve attempts to generalize the standard logical connectives and quantifiers to create combining rules for probabilities encountered many difficulties. Graphical probability models have become popular as a parsimonious language for representing knowledge about uncertain phenomena, a formalism for representing probabilistic knowledge in a logically coherent manner, and an architecture to support efficient algorithms for inference, search, optimization, and learning. A graphical probability model expresses a probability distribution over a collection of related hypotheses as a graph and a collection of local probability distributions. The graph encodes dependencies among the hypotheses. The local probability distributions specify numerical probability information. Together, the graph and the local distributions specify a joint distribution that respects the conditional independence assertions encoded in the graph, and has marginal distributions consistent with the local distributions (Cowell, et al., 1999; Jensen, 2001; Lauritzen, 1996; Pearl, 1988; Whittaker, 1990). A Bayesian network (e.g., Pearl, 1988; Jensen, 2001; Neapolitan, 2003) is a graphical probability model in which the dependency graph is an acyclic directed graph. An example of a Bayesian network for a diagnostic task is given in Figure 1. This Bayesian network represents a joint distribution over the Cartesian product of the possible values of the random variables depicted in the graph.

Some authors assume that random variables in a Bayesian network have finitely many possible values. Some require only that each random variable have an associated function mapping values of its parents to probability distributions on its set of possible values. In an unconstrained local distribution on finite-cardinality random variables, a separate probability is specified for each value of a random variable given each combination of values of its parents. Because the complexity of specifying local distributions is exponential in the number of parents, constrained families of local distributions are often used to simplify specification and inference. Examples include context-specific independence (Geiger and Heckerman, 1991; Boutilier, et al., 1996; Mahoney and Laskey, 1999; Mahoney, 1999) and independence of causal influence (ICI) models such as the "noisy or" (Jensen, 2001; Pearl, 1988). When a random variable and/or its

parents have infinitely many possible values, local distributions cannot be listed explicitly, but can be specified as parameterized functions using local expression languages (D'Ambrosio, 1991). When a random variable has an uncountable set of possible values, then the local distributions specify probability density functions with respect to a measure on the set of possible outcomes (cf., Billingsley, 1995; DeGroot and Schervish, 2002).

The simple attribute-value representation of standard Bayesian networks is insufficiently expressive for many problems. For example, the Bayesian network of Figure 1 applies to a single piece of equipment located in a particular room and owned and maintained by a single organization. We may need to consider problems that involve multiple organizations, each of which owns and maintains multiple pieces of equipment of different types, some of which are in rooms that contain other items of equipment. The room temperature and air conditioner status random variables would have the same value for co-located items, and the maintenance practice random variable would have the same value for items with the same owner. Standard Bayesian networks provide no way of compactly representing the correlation between failures of co-located and/or commonly owned items of equipment or of properly accounting for these correlations when learning from observation. For this reason, extensions to the Bayesian network formalism have been developed to provide greater expressivity.

Object-oriented Bayesian networks (OOBNs) and probabilistic relational models (PRMs) provide a natural way to represent uncertainty about the attributes of instances of different types of objects, where objects of a given type have attributes drawn from the same distribution (c.f., Pfeffer, 2000). That is, they provide a direct way to represent uncertainty about the values of unary functions and relations. Representing uncertainty about *n*-ary functions and relations is more cumbersome. One must reify an *n*-element argument sequence, define an attribute of the reified entity to represent the desired function, and then specify a distribution for the attribute.

Unlike OOBNs and PRMs, logic-based languages (e.g., Ngo and Haddawy, 1997) can represent *n*-ary functions and relations in a straightforward way. Whereas the unit of expression for OOBNs and PRMs is an object type and its attributes, the unit of expression for logic-based languages is the local distribution for an individual term in the language.

Like logic-based languages, MEBN can represent uncertainty about the values of *n*-ary functions and relations in a natural way. An attractive feature of MEBN is that distributions are specified over conceptually meaningful clusters of related hypotheses. This unit of representation facilitates modular specification of MFrag knowledge bases. Unlike PRMs or OOBNs, the hypotheses represented in a given MFrag need not be attributes of a single entity or a single reified list of entities, but can refer to attributes and relations of different entities. The arguments of random variables are defined at the level of the MFrag. Therefore, when an MFrag is instantiated, all occurrences of a given argument must be bound to the same entity. This constraint is useful in some applications, and is natural to specify in MEBN, but is cumbersome to represent in other languages. In these respects, MEBN is similar to the plate language (Gilks, et al., 1994), another language for which the unit of representation is a cluster of related random variables. However, plates cannot express nested function application or uncertainty about existence, type, and number. These kinds of uncertainty are easily expressed in MEBN.

The most natural semantics for first-order probabilistic languages is an extension of first-order model theory that assigns probabilities to sets of interpretations (cf., Russell and Norvig, 2002, Section 14.6). All the above formalisms can all be given this kind of declarative semantics. That is, a probabilistic knowledge base for any of the above formalisms can be translated into: (*i*) a set of first-order axioms defining a set of possible worlds; (*ii*) statements that allow probabilities

to be assigned in a consistent way to sets of possible worlds; and optionally (*iii*) context axioms that index probability distributions. A probabilistic knowledge base then specifies a probability distribution on possible worlds, or in some languages a family of probability distributions indexed by contexts. Different languages have the power to express different subsets of first-order logic, with different implications for tractable inference in particular classes of problem.

Development of probabilistic languages is a vital area of research. A range of solutions is needed to balance expressiveness against tractability. Among the variety of approaches, some are needed that push the bounds of expressive power outward. Highly expressive languages are necessary for knowledge interchange, for analyzing the theoretical properties of languages with different expressive power, for representing arbitrarily complex knowledge bases and ontologies in machine-understandable form, and for defining tractable special cases and approximations to intractable or undecidable problems. Languages and implementations based on subsets of first-order logic vary widely in expressiveness and tractability for different problem classes. As an example of a very expressive logic, the newly released Common Logic standard (ISO/IEC 2007) is intended for exchange of information among diverse applications. As such, one of its requirements is to be at least as expressive as first-order logic. This implies, of course, that it can represent intractable and even undecidable problems. Although this renders Common Logic an undesirable representation medium for some applications, this level of expressiveness is necessary for other purposes.

In a similar vein, there is a need for formalisms capable of specifying arbitrarily expressive probabilistic knowledge bases. For this reason, MEBN was designed to be capable of representing arbitrary first-order sentences. Theorem 5 of Section 4.2 proves that MEBN can represent a probability distribution over interpretations of any finitely axiomatizable first-order theory. This is a non-trivial result. Because sets of first-order axioms may have uncountably many interpretations, simply defining a probability distribution over interpretations does not ensure that the set of models of an arbitrary sentence will be measurable. In other words, there is no guarantee that an arbitrary probability distribution on first-order sentences will assign a probability to every sentence. Furthermore, because first-order logic is undecidable, inference in any probabilistic logic that contains first-order logic must be undecidable in the worst case. Nevertheless, MEBN can implicitly specify answers to arbitrary probabilistic queries about interpretations of finitely axiomatizable theories, and the process defined in Section 4.3 converges to the correct answer in the infinite limit. These results are theoretically significant.

Theoretical issues aside, practical applications demand tractable solutions. The common semantics shared by MEBN and other first-order probabilistic languages facilitates the definition of translations among representations (cf., Heckerman, et al., 2004). This makes it possible to define restricted subsets of MEBN for which tractable solutions exist to certain classes of problems. MEBN implementations may make restrictions on expressivity to ensure tractability. Knowledge engineers and inference algorithm designers may make various trade-offs to ensure acceptable performance in applications. Section 5 provides a brief discussion of the relationship between MEBN and other expressive probabilistic languages. The discussion in that section could be extended to establish translations from other expressive first-order languages into MEBN, and from appropriately restricted variants of MEBN into other languages.

3 Multi-Entity Bayesian Networks

Like Bayesian networks, MEBN theories use directed graphs to specify joint probability distributions for a collection of related random variables. The MEBN language extends ordinary

Bayesian networks to provide first-order expressive power, and also extends first-order logic (FOL) to provide a means of specifying probability distributions over interpretations of first-order theories.

Knowledge in MEBN theories is expressed via *MEBN Fragments* (MFrags), each of which represents probability information about a group of related random variables. Just as first-order logic extends propositional logic to provide an inner structure for sentences, MEBN theories extend ordinary Bayesian networks to provide an inner structure for random variables. Random variables in MEBN theories take arguments that refer to entities in the domain of application. For example, *Manager*(*d*,*y*) might represent the manager of the department designated by the variable *d* during the year designated by the variable *y*. To refer to the manager of the maintenance department in 2003, we would fill in values for *d* and *y* to obtain an instance *Manager*(*Maintenance*,2003) of the *Manager* random variable. A given situation might involve any number of instances of the *Manager* random variable, referring to different departments and/or different years. As shown below, the Boolean connectives and quantifiers of first-order logic are represented as pre-defined MFrags whose meaning is fixed by the semantics. A MEBN theory implicitly expresses a joint probability distribution over truth-values of sets of FOL sentences. Any sentence that can be expressed in first-order logic can be represented as a random variable in a MEBN theory. The MEBN language is modular and compositional. That is, probability distributions are specified locally over small groups of hypotheses and composed into globally consistent probability distributions over sets of hypotheses.

3.1 Entities and Random Variables

The MEBN language treats the world as being comprised of entities that have attributes and are related to other entities. Constant and variable symbols are used to refer to entities. There are three logical constants with meaning fixed by the semantics of the logic, an infinite collection of variable symbols, and an infinite collection of domain-specific constant symbols with no pre-specified referents. Random variables represent features of entities and relationships among entities. There is a collection of logical random variable symbols with meaning fixed by the semantics of the logic, and an infinite collection of domain-specific random variable symbols with no pre-specified referents. The logical constants and random variables are common to all MEBN theories; the domain-specific constants and random variables provide terminology for referring to objects and relationships in a domain of application.

Constant and variable symbols:

- (*Ordinary*) *variable symbols*: As in FOL, variables are used as placeholders to refer to non-specific entities. Variables are written as alphanumeric strings beginning with lowercase letters, e.g., *department*7. To avoid confusion, the adjective "ordinary" is sometimes used to distinguish ordinary variables from random variables.
- *Phenomenal (non-logical) constant symbols*: Particular named entities are represented using constant symbols. As in our FOL notation, phenomenal constant symbols are written as alphanumeric strings beginning with uppercase letters, e.g., *Machine*37, *Fernandez*.
- *Unique Identifier symbols*: The same entity may be represented by different phenomenal constant symbols. MEBN avoids ambiguity by assigning a unique identifier symbol to each entity. The unique identifiers are the possible values of random variables. There are two kinds of unique identifier symbols:

- *Truth-value symbols and the undefined symbol*: The reserved symbols T, F and ⊥ are logical constants with pre-defined meaning fixed by the semantics. The symbol ⊥ denotes meaningless, undefined or contradictory hypotheses, i.e., hypotheses to which a truth-value cannot be assigned. The symbols T and F denote truth-values of meaningful hypotheses.
- *Entity identifier symbols.* There is an infinite set $\mathcal{E}$ of entity identifier symbols. An interpretation of the theory uses entity identifiers as labels for entities. Entity identifiers are written either as numerals or as alphanumeric strings beginning with an exclamation point, e.g., !*M*3, 48723.

Random variable symbols:

- *Logical connectives and the equality operator*: The logical connective symbols ¬, ∧, ∨, ⇒, and ⇔, together with the equality relation =, are reserved random variable symbols with pre-defined meanings fixed by the semantics. Logical expressions may be written using prefix notation (e.g,, $\neg(\psi)$, $\vee(\psi,\phi)$, $=(\psi,\phi)$), or in the more familiar infix notation (e.g., $\neg\psi$, $(\psi\vee y)$; $(\psi=y)$). Different ways of writing the same expression (e.g., $=(\psi,\phi)$, $(\phi=\psi)$) are treated as the same random variable.
- *Quantifiers*: The symbols ∀ and ∃ are reserved random variable symbols with pre-defined meaning fixed by the semantics. They are used to construct MEBN random variables to represent FOL sentences containing quantifiers.
- *Identity*: The reserved random variable symbol ◊ denotes the identity random variable. It is the identity function on T, F, ⊥, and the set of entity identifiers that denote meaningful entities in a domain. It maps meaningless, irrelevant, or contradictory random variable terms to ⊥.
- *Findings*: The *finding* random variable symbol, denoted Φ, is used to represent observed evidence, and also to represent constraints assumed to hold among entities in a domain of application.
- *Domain-specific random variable symbols*: The domain-specific random variable symbols are written as alphanumeric strings beginning with an uppercase letter. With each random variable symbol is associated a positive integer indicating the number of arguments it takes. Each random variable also has an associated set of *possible values* consisting of a recursive subset of the unique identifier symbols. The set of possible values may be infinite, but if so, there must exist an effective procedure that lists all the possible values and an effective procedure for determining whether any unique identifier symbol is one of the possible values. If the set of possible values is contained in {T,F,⊥}, the random variable is called a *logical* random variable. For all other random variables, called *phenomenal* random variables, the set of possible values is contained in $\mathcal{E}\cup\{\perp\}$. Logical random variables correspond to predicates and phenomenal random variables correspond to functions in FOL. Local distributions (see Definition 3 below) may further restrict the set of possible values as a function of the values of the random variable's parents.
- *Exemplar symbols*. There is an infinite set of exemplar symbols used to refer to representative fillers for variables in the range of quantifiers. An exemplar symbol is denoted by $ followed by an alphanumeric string, e.g., $*b*32.[1]

[1] Exemplar symbols were called Skolem symbols in earlier work (e.g., Laskey and Costa, 2005) because, in analogy to Skolem functions, exemplar symbols replace variables in the range of quantifiers. However, exemplars are different from Skolem functions, and the terminology was changed to avoid confusion.

Punctuation:

- MEBN random variable terms are constructed using the above symbols and the punctuation symbols comma, open parenthesis and close parenthesis.

A *random variable term* is a random variable symbol followed by a parenthesized list of arguments separated by commas, where the arguments may be variables, constant symbols, or (recursively) random variable terms. When α is a constant or ordinary variable, the random variable term $\Diamond(\alpha)$ may be denoted simply as α. If ψ is a random variable symbol, a *value assignment term* for ψ has the form $=(\psi,\alpha)$, where ψ is a random variable term and α is either an ordinary variable symbol or one of the possible values of ψ. The strings $=(\alpha,\psi)$, $(\alpha=\psi)$, and $(\psi=\alpha)$ are treated as synonyms for $=(\psi,\alpha)$. A random variable term is *closed* if it contains no ordinary variable symbols and *open* if it contains ordinary variable symbols. An open random variable term is also called a *random variable class*; a closed random variable term is called a *random variable instance*. If a random variable instance is obtained by substituting constant terms for the variable terms in a random variable class, then it is called an instance of the class. For example, the value assignment term =(*BeltStatus*(!*B*1), !*OK*), also written (*BeltStatus*(!*B*1) = !*OK*), is an instance of both (*BeltStatus*(*b*)=*x*) and (*BeltStatus*(!*B*1)=*x*), but not of (*BeltStatus*(*b*) = !*Broken*). When no confusion is likely to result, random variable classes and instances may be referred to as random variables. A random variable term is called *simple* if all its arguments are either unique identifier symbols or variable symbols; otherwise, it is called *composite*. For example, =(*BeltStatus*(!*B*1), !*OK*) is a composite random variable term containing the simple random variable term *BeltStatus*(!*B*1) as an argument. It is assumed that the sets consisting of ordinary variable symbols, unique identifier symbols, exemplar random variable symbols, phenomenal constant symbols, and domain-specific random variable symbols are all recursive.

3.2 MEBN Fragments

In MEBN theories, multivariate probability distributions are built up from *MEBN fragments* or MFrags (see Figure 2). An MFrag defines a probability distribution for a set of *resident* random variables conditional on the values of *context* and *input* random variables. Random variables are represented as nodes in a fragment graph whose arcs represent dependency relationships.

***Definition 1*:** An *MFrag* $\mathcal{F} = (\mathcal{C},\mathcal{I},\mathcal{R},\mathcal{G},\mathcal{D})$ consists of a finite set $\mathcal{C}$ of *context* value assignment terms;[2] a finite set $\mathcal{I}$ of *input* random variable terms; a finite set $\mathcal{R}$ of *resident* random variable terms; a *fragment graph* $\mathcal{G}$; and a set $\mathcal{D}$ of *local distributions*, one for each member of $\mathcal{R}$. The sets $\mathcal{C}$, $\mathcal{I}$, and $\mathcal{R}$ are pairwise disjoint. The fragment graph $\mathcal{G}$ is an acyclic directed graph whose nodes are in one-to-one correspondence with the random variables in $\mathcal{I}\cup\mathcal{R}$, such that random variables in $\mathcal{I}$ correspond to root nodes in $\mathcal{G}$. Local distributions specify conditional probability distributions for the resident random variables as described in Definition 3 below. ■

An MFrag is a schema for specifying conditional probability distributions for instances of its resident random variables given the values of instances of their parents in the fragment graph and given the context constraints. A collection of MFrags that satisfies the global consistency constraints defined in Section 3.3 below represents a joint probability distribution on an unbounded and possibly infinite number of instances of its random variable terms. The joint distribution is specified via the local distributions, which are defined formally below, together

[2] If ϕ is a Logical random variable, the context constraint ϕ=T may be abbreviated ϕ and the context constraint ϕ=F may be abbreviated $\neg\phi$.

with the conditional independence relationships implied by the fragment graphs. Context terms are used to specify constraints under which the local distributions apply.

As in ordinary Bayesian networks, a local distribution maps configurations of values of the parents of a random variable instance to probability distributions for its possible values. When all ordinary variables in the parents of a resident random variable term also appear in the resident term itself, as for the *RoomTemp* and *TempLight* random variables of the temperature observability MFrag of Figure 2, a local distribution can be specified simply by listing a probability distribution for the child random variable for each combination of values of the parent random variables. The situation is more complicated when ordinary variables in a parent random variable do not appear in the child. In this case, there may be an arbitrary, possibly infinite number of instances of a parent for any given instance of the child. For example, in the engine status fragment of Figure 2, if it is uncertain where a machine is located, the temperature in any room in which it might be located is relevant to the distribution of the *EngineStatus* random variable. If a machine has more than one belt, then the status of any of its belts is relevant to the distribution of the *EngineStatus* random variable. Thus, any number of instances of the *RoomTemp* and *BeltStatus* random variables might be relevant to the distributions of the *EngineStatus* random variable. In this case, the local distribution for a random variable must specify how to combine influences from all relevant instances of its parents. The standard approaches to this problem are aggregation functions and combining rules (cf., Natarajan, et al., 2005).

MEBN local distributions combine influences of multiple parents through *influence counts*. In a standard Bayesian network, the probability distribution for a node depends on the configuration of states of its parents. In a MEBN theory, different substitutions for the ordinary variables may yield multiple instantiations of the parents. Each allowable substitution defines a parent set, and each parent set has a configuration of states. Influence counts tally the number of times each configuration of the parents occurs among these parent sets.

Influence counts may seem unintuitive at first, but they extend the causal Markov condition of ordinary Bayesian networks in a natural and very general way. According to the causal Markov condition, the distribution of a child may depend on no information except the values taken on by its parents. For a Bayesian network, each node has a fixed number of parents that have exactly one configuration in any given possible world. The local distribution in a Bayesian network is a function of this configuration. For an MFrag, when there are multiple instances of the parents of a random variable, this gives rise to a set of parent configurations, one for each allowable substitution for the arguments of the parents. The basic idea of influence counts is that when multiple bindings result in the same configuration of parent states, the only information relevant to the local distribution should be how many cases there are of each configuration, not any other information such as which particular instances participated in each of the bindings. This is exactly what is represented by influence counts. In the engine status example, if a machine might be located in one of several rooms and might have more than one belt, then there is a configuration of the *RoomStatus* and *RoomTemp* variables for each allowable substitution of rooms and belts. This is explained in detail in the example below.

Configurations of the parent random variables that are relevant to the distribution of the child are called *influencing configurations*. The local distribution π_ψ for a resident random variable ψ in MFrag $\mathcal{F}$ specifies, for each instance of ψ: (*i*) a set of *possible values*; (*ii*) a rule for determining the influencing configurations; and (*iii*) a rule for assigning probabilities to the possible values given an influencing configuration. This statement is formalized in Definition 3 below. Before

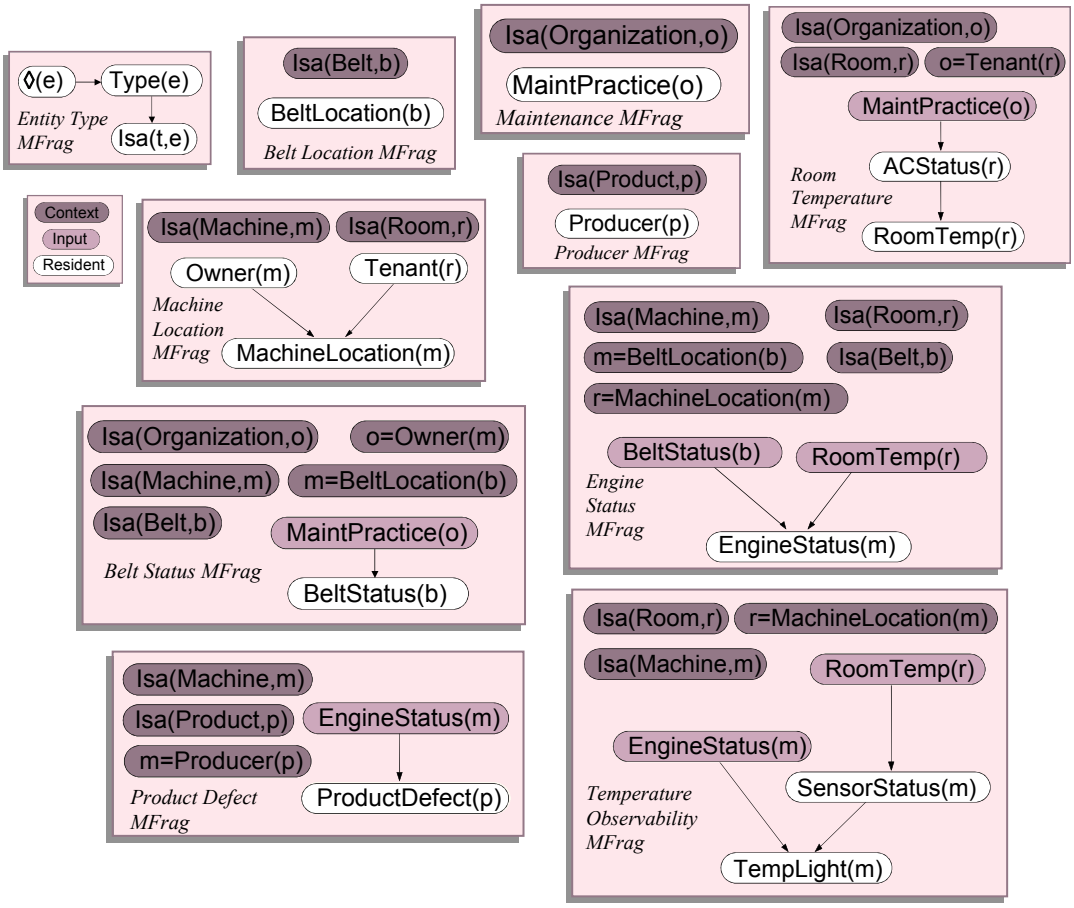


Figure 2: MEBN Fragments for Equipment Diagnosis Problem

proceeding to a formal definition of local distributions, a formal definition of influence counts is given, followed by an example of how they are used to define local distributions.

***Definition 2*:** Let $\mathcal{F}$ be an MFrag containing ordinary variables θ_1, …, θ_k, and let $\psi(\theta)$ denote a resident random variable in $\mathcal{F}$ that may depend on some or all of the θ_i.

2a. A *binding set* $\mathcal{B} = \{(\theta_1:\varepsilon_1), (\theta_2:\varepsilon_2), \ldots (\theta_k:\varepsilon_k)\}$ for $\mathcal{F}$ is a set of ordered pairs associating a unique identifier symbol ε_i with each ordinary variable θ_i of $\mathcal{F}$. The constant symbol ε_i is called the *binding* for variable θ_i determined by $\mathcal{B}$. The ε_i are not required to be distinct.

2b. Let $\mathcal{B} = \{(\theta_1:\varepsilon_1), (\theta_2:\varepsilon_2), \ldots (\theta_k:\varepsilon_k)\}$ be a binding set for $\mathcal{F}$, and let $\psi(\varepsilon)$ denote the instance of ψ obtained by substituting ε_i for each occurrence of θ_i in $\psi(\theta)$. A *potential influencing configuration* for $\psi(\varepsilon)$ and $\mathcal{B}$ is a set of value assignment terms $\{(\gamma=\phi(\varepsilon))\}$, one for each parent of ψ and one for each context random variable of $\mathcal{F}$. Here, $\phi(\varepsilon)$ denotes the instance of the context or parent random variable $\phi(\theta)$ obtained by substituting ε_i for each occurrence of θ_i;[3] and γ denotes one of the possible values of $\phi(\varepsilon)$ (as specified by the local distribution π_ϕ; see Definition 3 below). An *influencing configuration* for $\psi(\varepsilon)$ and $\mathcal{B}$ is a potential influencing configuration in which the value assignments match the context constraints of $\mathcal{F}$. Two influencing configurations are

[3] If a context value assignment term ($\gamma=\phi$) has no arguments, then no substitution is needed.

equivalent if substituting θ_i back in for ε_i yields the same result for both configurations. The equivalence classes for this equivalence relation correspond to distinct configurations of parents of $\psi(\theta)$ in $\mathcal{F}$.

2c. Let $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$ be a non-empty, finite set of entity identifier symbols. The *partial world* $\mathcal{W}$ for ψ and $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$ is the set consisting of all instances of the parents of ψ and the context random variables of $\mathcal{F}$ that can be formed by substituting the ε_i for ordinary variables of $\mathcal{F}$. A *partial world state* $S_{\mathcal{W}}$ for a partial world is a set of value assignment terms, one for each random variable in the partial world.

2d. Let $\mathcal{W}$ be a partial world for ψ and $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$, let $S_{\mathcal{W}}$ be a partial world state for $\mathcal{W}$, let $\mathcal{B} = \{(\theta_1{:}\varepsilon_{\mathcal{B}1}), (\theta_2{:}\varepsilon_{\mathcal{B}2}), \ldots (\theta_k{:}\varepsilon_{\mathcal{B}k})\}$ be a binding set for $\mathcal{F}$ with bindings chosen from $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n\}$, and let $\psi(\varepsilon_{\mathcal{B}})$ be the instance of $\psi(\theta)$ from $\mathcal{B}$. The *influence counts* $\#S_{\mathcal{W}\psi}$ for $\psi(\alpha_{\mathcal{B}})$ in $S_{\mathcal{W}}$ consist of the number of influencing configurations $S_{\mathcal{W}}$ contains for each equivalence class of influencing configurations (i.e., each configuration of the parents of $\psi(\theta)$ in $\mathcal{F}$). ■

As an example, Table 1 shows a partial world state for the *EngineStatus*(*m*) random variable from Figure 2 with unique identifiers {!*M*1, !*R*1, !*R*2, !*B*1, !*B*2, !*O*1}. In the intended meaning of the partial world of Table 1, !*M*1 denotes a machine, !*B*1 and !*B*2 denote belts located in !*M*1, !*R*1 denotes the room where !*M*1 is located, !*R*2 denotes a room where !*M*1 is not located, and !*O*1 denotes an entity that is not a machine, a room, or a belt. The partial world state specifies the value of each random variable for each of the entity identifiers. Random variables map meaningless attributes (e.g., the value of *RoomTemp* for an entity that is not a room) to the absurd symbol ⊥.

To construct the influencing configurations, we first examine Table 1 to find all configurations of context random variables that satisfy the context constraints. The first constraint is that the entity bound to *m* must be a machine. The only instance of *Isa*(*Machine*, *m*) with value T binds *m* to !*M*1. Therefore, all influencing configurations must include (*Isa*(*Machine*,!*M*1)=T). Next, consider the constraint that the entity bound to *r* must be a room. There are two assignments satisfying this constraint: (*Isa*(*Room*,!*R*1)=T) and (*Isa*(*Room*,!*R*2)=T). All

Isa(*Machine*,!*M*1)=T	*Isa*(*Machine*,!*R*1)=F	*Isa*(*Machine*,!*R*2)=F
Isa(*Belt*,!*M*1)=F	*Isa*(*Belt*,!*R*1)=F	*Isa*(*Belt*,!*R*2)=F
Isa(*Room*,!*M*1)=F	*Isa*(*Room*,!*R*1)=T	*Isa*(*Room*,!*R*2)=T
BeltLocation(!*M*1)=⊥	*BeltLocation*(!*R*1)=⊥	*BeltLocation*(!*R*2)=⊥
MachineLocation(!*M*1)=!*R*1	*MachineLocation*(!*R*1)=⊥	*MachineLocation*(!*R*2)=⊥
RoomTemp(!*M*1)=⊥	*RoomTemp*(!*R*1)=!*Normal*	*RoomTemp*(!*R*2)=*Hot*
BeltStatus(!*M*1)=⊥	*BeltStatus*(!*R*1)=⊥	*BeltStatus*(!*R*2)=⊥
Isa(*Machine*,!*B*1)=F	*Isa*(*Machine*,!*B*2)=F	*Isa*(*Machine*,!*O*1)=F
Isa(*Belt*,!*B*1)=T	*Isa*(*Belt*,!*B*2)=T	*Isa*(*Belt*,!*O*1)=F
Isa(*Room*,!*B*1)=F	*Isa*(*Room*,!*B*2)=F	*Isa*(*Room*,!*O*1)=F
BeltLocation(!*B*1)=!*M*1	*BeltLocation*(!*B*2)=!*M*1	*BeltLocation*(!*O*1)=⊥
MachineLocation(!*B*1)=⊥	*MachineLocation*(!*B*2)=⊥	*MachineLocation*(!*O*1)=⊥
RoomTemp(!*B*1)=⊥	*RoomTemp*(!*B*2)=⊥	*RoomTemp*(!*O*1)=⊥
BeltStatus(!*B*1)=!*OK*	*BeltStatus*(!*B*2)=!*OK*	*BeltStatus*(!*O*1)=⊥

Table 1: Partial World State for *EngineStatus* Partial World

influencing configurations must contain one of these two value assignments. Now, consider the third context constraint, that the machine bound to *m* must be located in the room bound to *r*. This constraint is satisfied by only one binding, of *m* to !*M*1 and *r* to !*R*1. This eliminates (*Isa*(*Room*,!*R*2)=T) from the influencing configurations. Thus, all influencing configurations must contain (*Isa*(*Room*,!*R*1)=T) and (*MachineLocation*(!*M*1)=!*R*1). Next, consider the constraints that the entity bound to *b* must be a belt located in the machine bound to *m*. These constraints are satisfied by binding *b* to either !*B*1 or !*B*2. Therefore, each influencing configuration must contain either (*Isa*(*Belt*,!*B1*)=T) and (*BeltLocation*(!*B1*)=!*M*1), or (*Isa*(*Belt*,!*B2*)=T) and (*BeltLocation*(!*B2*)=!*M*1). Putting all this information together, the partial world state of Table 1 contains two influencing configurations for *EngineStatus*(!*M*1):

IC1: { (*Isa*(*Machine*,!*M*1)=T), (*Isa*(*Belt*,!*B1*)=T), (*Isa*(*Room*,!*R*1)=T), (*BeltLocation*(!*B1*)=!*M*1), (*MachineLocation*(!*M*1)=!*R*1), (*RoomTemp*(!*R*1)=!*Normal*), (*BeltStatus*(!*B*1)=!*OK*)}; and

IC2: { *Isa*(*Machine*,!*M*1)=T), (*Isa*(*Belt*,!*B*2)=T), (*Isa*(*Room*,!*R*1)=T), (*BeltLocation*(!*B*2)=*M*1), (*MachineLocation*(!*M*1)=!*R*1), (*RoomTemp*(!*R*1)=!*Normal*), (*BeltStatus*(!*B*2)=!*OK*)}.

The partial world state of Table 1 contains no other influencing configurations for *EngineStatus*(!*M*1). In both IC1 and IC2, the room temperature is normal and the belt status is OK. Therefore, the influence counts for *EngineStatus*(!*M*1) in this possible world state are:

RoomTemp=!*Normal*,	*BeltStatus*=!*OK*	:	2
RoomTemp=!*Normal*,	*BeltStatus*=!*Broken*	:	0
RoomTemp=!*Hot*,	*BeltStatus*=!*OK*	:	0
RoomTemp=!*Hot*,	*BeltStatus*=!*Broken*	:	0 .

The local distribution assigned to *EngineStatus*(*M*1) in this partial world state would thus be the one for a machine having two intact and no broken belts, and located in a room with normal room temperature.

Because of their generality, influence counts can represent both aggregation functions and combining rules (see discussion in Section 5 below). Implementations of MEBN are free to restrict local distributions for reasons of tractability or cognitive naturalness. For example, an implementation might provide a set of standard combining rules such as noisy Boolean and arithmetic functions, and possibly also provide a language for defining combining functions, but would not necessarily provide a fully general language for specifying local distributions as a function of influence counts. That is, influence counts provide for the most general specification of local distributions, but this full generality would not necessarily be available in every implementation.

***Definition 3*:** The *local distribution* π_ψ for resident random variable ψ in MFrag $\mathcal{F}$ specifies, for each instance $\psi(\varepsilon)$ of ψ: (*i*) a subset $\mathcal{V}_{\psi(\varepsilon)}$ of *possible values* for $\psi(\varepsilon)$; and (*ii*) a function $\pi_{\psi(\varepsilon)}(\alpha|S)$ that maps unique identifiers α and partial world states *S* to real numbers, such that the following conditions are satisfied:

3a. For a given partial world state *S*, $\pi_{\psi(\varepsilon)}(\cdot|S)$ is a probability distribution on the unique identifier symbols. That is, $\pi_{\psi(\varepsilon)}(\alpha|S) \geq 0$ and $\sum_\alpha \pi_{\psi(\varepsilon)}(\alpha \mid S) = 1$, where α ranges over the unique identifier symbols.

3b. For each instance $\psi(\varepsilon)$ of ψ, the set $\mathcal{V}_{\psi(\varepsilon)}$ of possible values of the instance $\psi(\varepsilon)$ is a recursive subset of the set of possible values for ψ, and $\pi_{\psi(\varepsilon)}(\mathcal{V}_{\psi(\varepsilon)}|S) = 1$ for each partial world S. As noted above, there is a set of possible values associated with the random variable ψ. This condition states that the set of possible values for an instance $\psi(\varepsilon)$ may be a proper subset of the set of possible values for ψ. If this is the case, then the parents of ψ in the fragment graph must include the entity identifiers for one or more of the arguments of ψ, and the possible values of an instance $\psi(\varepsilon)$ must be a function of the entity identifiers $\Diamond(\varepsilon_i)$ of one or more arguments ε_i. That is, the set of possible values for an instance $\psi(\varepsilon)$ may depend only on the random variable class ψ and the identifiers of the entities being substituted for its arguments.

3c. There is an algorithm such that for any finite subset A of the possible values of $\psi(\varepsilon)$ not containing $\perp$, and for any partial world state S for ψ, either the algorithm halts with output $\pi_{\psi(\varepsilon)}(A|S)$ or there exists a value $N(A,S)$ such that if the algorithm is interrupted after a number of time steps greater than $N(A,S)$, the output is $\pi_{\psi(\varepsilon)}(A|S)$.[4]

3d. $\pi_{\psi(\varepsilon)}$ depends on the partial world state only through the influence counts. That is, any two partial world states having the same influence counts map to the same probability distribution;

3e. Let $S_1 \subset S_2 \subset \ldots$ be an increasing sequence of partial world states for ψ, and let α be one of the possible values for ψ. There exists an integer N such that if $k > N$, $\pi_{\psi(\varepsilon)}(\alpha|S_k) = \pi_{\psi(\varepsilon)}(\alpha|S_N)$.[5]

The probability distribution $\pi_{\psi}(\varepsilon|\varnothing)$ is called the *default distribution* for ψ. It is the probability distribution for ψ given that no potential influencing configurations satisfy the conditioning constraints of $\mathcal{F}$. If ψ is a root node in an MFrag $\mathcal{F}$ containing no context constraints, then the local distribution for ψ is just the default distribution. ■

Conditions such as 3*c* and 3*e* are needed to ensure that a global joint distribution exists and can be approximated by a sequence of finite Bayesian networks. The conditions given here are stronger than strictly necessary. Because they are satisfied in the MEBN theory for first-order logic presented in Section 4.2 below, they are sufficient to demonstrate the existence of a fully first-order Bayesian logic. Nevertheless, identifying suitable relaxations of these conditions is an important topic for future research. For example, in some applications it would be useful to define a random variable as the average of infinitely many instances of its parent. It is clear that such a local distribution would not satisfy Condition 3*e*. Results on convergence of averages to limiting distributions (e.g., Billingsley, 1995) might be applied to identify suitable relaxations of these conditions. It should be noted in this connection that most papers on expressive probabilistic languages explicitly assume the domain is finite (see, for example, Heckerman, 2004). In finite domains, including finite domains of uncertain and unbounded cardinality, conditions 3*c* and 3*e* are automatically satisfied.

Although the sets $\mathcal{V}_{\psi(\varepsilon)}$ are finite or countably infinite, it is possible to use MEBN to define distributions on arbitrary spaces. We can view the entity identifiers as labels for the elements of a sequence sampled randomly from a set that may be uncountably infinite. The characteristics of the sampled elements are specified via the distributions of features. For example, *StdUniform*(1), *StdUniform*(2), …, might represent labels for uniform random numbers drawn from the unit interval. We might define these labels as *StdUniform*(1) = !*StdUniform*1, *StdUniform*(2) =

[4] It is required that $N(A,S)$ exists, but there need not be an effective procedure for computing it.

[5] Again, it is not required that there be an effective procedure for computing N.

!*StdUniform*2, …, respectively. The random variable *Digit*(*u*,*k*) might then denote the k^{th} digit of the n^{th} uniform random number. The values *Digit*(*u*,*k*) would be mutually independent with uniform distributions on the set {0, 1}. A uniform random number could be specified to arbitrary precision by drawing a sufficiently long sequence of digits.

Table 2 shows an example of a local distribution for the engine status MFrag. The conditioning constraints imply there can be at most one *RoomTemp* parent that satisfies the context constraint *MachineLocation*(*m*) = *r*. When this parent has value !*Normal*, probability $\alpha_{k,n}$ is assigned to !*Normal* and probability $1\text{-}\alpha_{k,n}$ is assigned to !*Overheated*, where *k* is the number of distinct *BeltStatus* parents having the value *OK*, out of a total of *n*>0 distinct *BeltStatus* parents. When the *RoomTemp* parent corresponding to *MachineLocation*(*m*) has value !*Hot*, the probability of a satisfactory engine is $\beta_{k,n}$ and the probability of an overheated engine is $1\text{-}\beta_{k,n}$, where again *k* denotes the number of distinct belts with value *OK* and *n*>0 denotes the total number of distinct belts. The default distribution applies when no combination of entities meets the conditioning constraints. It assigns probability 1 to ⊥, meaning that *EngineStatus*(*m*) is meaningless when the context constraints are not met (i.e., *m* does not denote a machine, *m* is not located in a room, or *m* has no belt). Default distributions need not assign probability 1 to ⊥. For example, the default distribution could be used to represent the engine status of beltless machines. Note, however, that the default distribution does not distinguish situations in which *m* refers to a machine with no belt from situations in which *m* is not a machine. Thus, this modeling approach would assign the same *EngineStatus* distribution to non-machines as to machines with no belt.

MFrags may contain recursive influences. Recursive influences allow instances of a random variable to depend directly or indirectly on other instances of the same random variable. One common type of recursive graphical model is a dynamic Bayesian network (Ghahramani, 1998; Murphy, 1998). Recursion is permissible as long as no random variable instance can directly or indirectly influence itself. This requirement is satisfied when the conditioning constraints prevent circular influences. For example, Figure 3 modifies the belt status MFrag from Figure 2 so that the status of a belt depends not only on the maintenance practice of the organization, but also on the status of the belt at the previous time. The context constraint *s* = *Prev*(*t*) prevents circular influences in instances of this MFrag. The distribution depends on a random variable *Prev*(*t*) whose home MFrag is not shown here. *Prev* maps each positive numeral to the previous numeral, and maps other entity identifiers to ⊥. If the variable *t* is bound to 0, there will be no potential influencing configurations that satisfy the context constraints (because *Prev*(0) has value ⊥ and *Isa*(*NatNumber*(⊥)=F). Therefore, by Definition 2, there are no influencing configurations for the *BeltStatus* random variable. Thus, any instance of the *BeltStatus* random variable for which *t* is

Context	***RoomTemp(r)***	***BeltStatus(b)***	***EngineStatus(m)***		
			Satisfactory	***Overheated***	⊥
Belt b located in machine m, located in room r	*!Normal*	*!OK : k*	$\alpha_{k,n}$	$1\text{-}\alpha_{k,n}$	0
		!Broken : n-k			
	!Hot	*!OK : k*	$\beta_{k,n}$	$1\text{-}\beta_{k,n}$	0
		!Broken : n-k			
Default			0	0	1

Table 2: Local Distribution as Function of Influence Counts

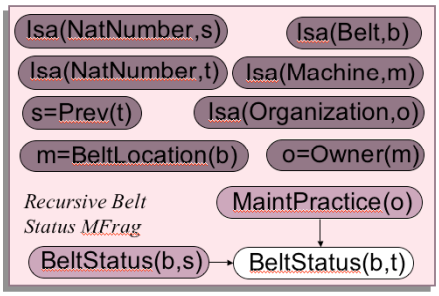


Figure 3: Recursive MFrag

bound to zero will have no parents, and its local distribution will be the default distribution. When t is bound to a positive numeral, the only influencing configuration will have s and t bound to consecutive numerals. Therefore, each *BeltStatus* instance has exactly one active *BeltStatus* parent, the one for the same belt at the previous time.

MFrags can represent a rich family of probability distributions over interpretations of first-order theories. The ability of MFrags to represent uncertainty about parameters of local distributions provides a logical foundation for parameter learning in first-order probabilistic theories. Uncertainty about structure can be represented by sets of MFrags having mutually exclusive context constraints and different fragment graphs, thus providing a logical foundation for structure learning. Of course, additional work is needed to define and implement learning algorithms for MEBN theories.

MEBN comes equipped with a set of built-in MFrags representing logical operations, function composition, and quantification. All other MFrags are called domain-specific MFrags. The domain-specific MFrags in a MEBN theory must satisfy constraints that ensure logical consistency of the theory. The built-in MFrags, the constraints on domain-specific MFrag definitions, and the rules for combining MFrags and performing inference provide the logical content of Bayesian logic. An applied MEBN theory augments the built-in MFrags with a set of domain-specific MFrags that provide empirical and/or mathematical content.

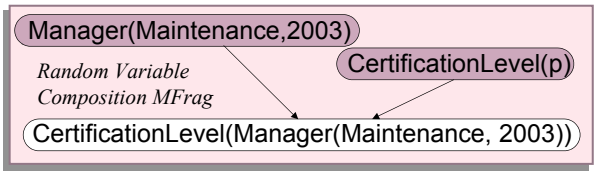


Figure 4: Indirect Reference

The built-in MFrags are defined below:

- *Indirect reference*. The rules for instantiating MFrags allow only unique identifier symbols to be substituted for the ordinary variable symbols. Probability distributions for indirect references are handled with built-in composition MFrags, as illustrated in Figure 4. These MFrags enforce logical constraints on function composition. Let $\psi(\phi_1(\alpha_1), \ldots, \phi_k(\alpha_k))$ be a random variable instance, where ψ and ϕ_i are random variable symbols and each α_i is a list of arguments. The random variable instance $\psi(\phi_1(\alpha_1), \ldots, \phi_k(\alpha_k))$ has a parent $\phi_i(\alpha_i)$ for each of the arguments and a *reference parent* $\psi(y_1, \ldots, y_k)$, where the y_i denote ordinary variable symbols such that y_i may be the same as y_j only if $\phi_i(\alpha_i)$ and $\phi_j(\alpha_j)$ are logically equivalent expressions.[6] The local distribution for $\psi(\phi_1(\alpha_1),\ldots,\phi_k(\alpha_k))$ assigns it the same value as $\psi(y_1,\ldots,y_k)$ when the value of y_i is the same as the value of $\phi_i(\alpha_i)$. Although there are infinitely many possible substitutions for $\psi(y_1,\ldots,y_k)$ and hence infinitely many potential influencing configurations, in any given world only one of the influences is active. Thus, condition 3*e* is satisfied. The default distribution specifies a value for $\psi(\phi_1(\alpha_1),\ldots,\phi_k(\alpha_k))$ when there are no influencing configurations.

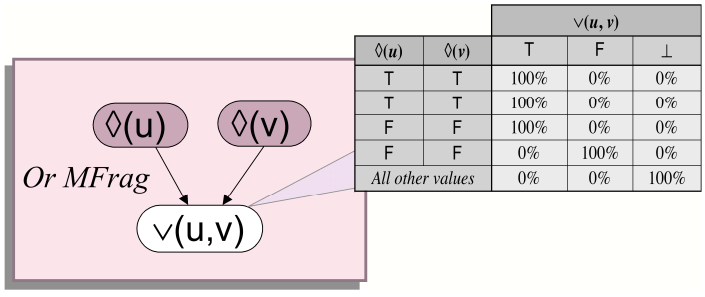
[6] It is always permissible to use distinct variables in a composition MFrag, but it is more efficient to use the same variable when the expressions are known to be logically equivalent.

- *Equality random variable*. The resident random variable in the equality MFrag has the form $=(u,v)$, also written $(u=v)$. There are two parents, one for each argument. The equality operator has value $\bot$ if either u or v has value $\bot$, T if ϕ and ψ have the same value and are not equal to $\bot$, and F otherwise. It is assumed that meaningful entity identifiers are distinct. That is, if ε_1 and ε_2 are distinct entity identifiers, then $(\varepsilon_1=\varepsilon_2)$ has value $\bot$ if $\Diamond(\varepsilon_1)$ or $\Diamond(\varepsilon_2)$ has value $\bot$, and F otherwise.
- *Logical connectives*. The random variable $\neg(u)$ has a single parent, $\Diamond(u)$; the other logical connectives have two parents, $\Diamond(u)$ and $\Diamond(v)$. The value of $\neg(u)$ is T if its parent has value F, F if its parent has value T, and $\bot$ otherwise. The other logical connectives map truth-values according to the usual truth tables and parents other than T or F to $\bot$ (see Figure 5).
- *Quantifiers*. Let $\phi(\gamma)$ be an open logical random variable term containing the ordinary variable γ. A *quantifier random variable* has the form $\forall(\sigma, \phi(\sigma))$ or $\exists(\sigma, \phi(\sigma))$, where $\phi(\sigma)$ is obtained by substituting the exemplar *term* σ into $\phi(\gamma)$. A quantifier random variable instance has a single parent $\phi(\gamma)$. The value of $\forall(\sigma, \phi(\sigma))$ is T by default and F if any instance of $\phi(\gamma)$ has value F. The value of $\exists(\sigma, \phi(\sigma))$ is F by default and T if any instance of $\phi(\gamma)$ has value T. It is assumed that a unique exemplar symbol is assigned to each ordinary variable of each logical random variable term of the language.[7] Figure 6 shows quantifier MFrags representing the hypothesis that every machine has a belt. In FOL, the corresponding sentence is:

$$\forall m \exists b\ (Isa(Machine,m) \Rightarrow Isa(Belt,b) \wedge (m=BeltLocation(b))).$$

An important feature of MEBN is its logically consistent treatment of reference uncertainty. For example, suppose the random variable instance *CertificationLevel*(*Manager*(*Maintenance*, 2003)) is intended to refer to the individual who managed the maintenance department in 2003. If the possible managers are *!Employee*37 and *!Employee*49, the distribution for *Certification-Level*(*Manager*(*Maintenance*, 2003)) will be a weighted average of the probability distributions for *CertificationLevel*(*!Employee*37) and *CertificationLevel*(*!Employee*49), where the weights are the probabilities that *Manager*(*Maintenance*, 2003) has value *!Employee*37 and *!Employee*49, respectively. If *!Employee*39 refers to an individual who is also referred to as *Carlos*, *Fernandez*, and *Father*(*Miguel*), any information germane to the certification level of *Carlos*, *Fernandez* or *Father*(*Miguel*) will propagate consistently to *CertificationLevel*(*Manager*(*Maintenance*, 2003)) when Bayesian inference is applied (see Figure 7). The indirect reference MFrags enforce these logical constraints on function composition.

The built-in MFrags defined above provide sufficient expressive power to represent a probability distribution over interpretations of any finitely axiomatizable FOL theory. Bayesian conditioning can be applied to generate a sequence of MEBN theories, where each theory in the sequence conditions the preceding theory on new axioms that are



		∨(u, v)		
◊(u)	◊(v)	T	F	⊥
T	T	100%	0%	0%
T	T	100%	0%	0%
F	F	100%	0%	0%
F	F	0%	100%	0%
All other values		0%	0%	100%

Figure 5: Logical Connective MFrag

[7] A countable infinity of exemplar symbols is sufficient for this purpose.

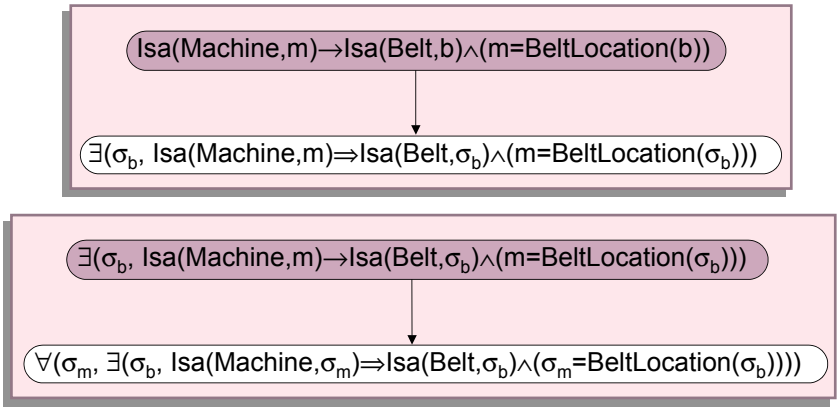


Figure 6: Quantifier MFrags

consistent with all previous axioms. MEBN theories can be used to define special-purpose logics such logics for planning and decision-making.

MEBN places no constraints on the distribution of exemplar constants. Implementations may treat them as constants that serve no purpose beyond their role as placeholders in quantifier random variables. They can, however, play an important role in modeling. An exemplar constant is intended as a label for a representative filler of its place in a quantifier random variable. Its distribution can be defined in a way that affects the probability that its associated sentence is satisfied. This is the role played by the exemplar distributions in the construction of Section 4.2. In this construction, exemplar constants for unsatisfiable sentences always have value ⊥, exemplar constants for valid sentences never have value ⊥, and exemplar constants for sentences that are neither valid nor unsatisfiable are assigned the value ⊥ with a probability strictly between 0 and 1. Assigning the value ⊥ to an exemplar constant constrains the generative distribution to ensure that the corresponding quantifier random variable cannot have value T. If a sentence is satisfiable, the exemplar for its negation has non-zero probability of being assigned value ⊥. When this occurs, the distribution defined in Section 4.2 will prevent its negation from being satisfied. Thus, the original sentence will have value T. Even if the joint distribution encoded by the other domain-specific MFrags would assign zero probability to T, the exemplars are sampled in a manner that ensures the sentence has positive chance of being satisfied. Details of this construction are provided in Section 4.2.

There are two kinds of domain-specific MFrags: generative MFrags and finding MFrags. The distinction between generative MFrags and finding MFrags corresponds roughly to the *terminological box*, or T-box, and the *assertional reasoner*, or A-box (Brachman, et al., 1983). The generative domain-specific MFrags specify information about statistical regularities characterizing the class of situations to which a MEBN theory applies. Findings can be used to specify particular information about a specific situation in the class defined by the generative theory. Findings can also be used to represent constraints assumed to hold in the domain (cf., Jensen,

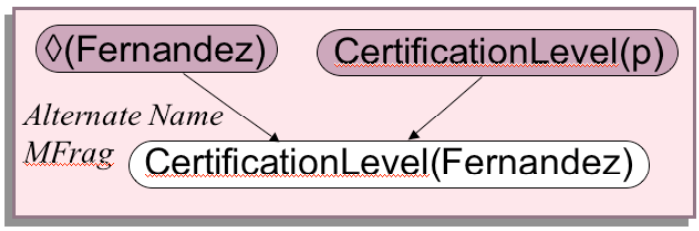


Figure 7: Relating a Name to a Unique Identifier

2001; Heckerman, et al., 2004), although there are both computational and interpretation advantages to specifying constraints generatively when possible. The two kinds of domain-specific MFrags are defined below.

***Definition* 4:** A *finding* MFrag satisfies the following conditions:

4a. There is a single resident random variable, $\Phi(\psi)$, where ψ is a closed value assignment term. For logical random variable instances, we may abbreviate $\Phi(\phi{=}\mathsf{T})$ as $\Phi(\phi)$, and $\Phi(\phi{=}\mathsf{F})$ as $\Phi(\neg(\phi))$.

4b. There are no context random variable terms. There is a single input random variable term ψ, which is a parent of the resident random variable $\Phi(\psi)$.

4c. The local distribution for $\Phi(\psi)$ is deterministic, assigning value T if ψ has value T and $\perp$ if it has value F or $\perp$. ■

***Definition* 5:** A generative domain-specific MFrag $\mathcal{F}$ must satisfy the following conditions.

5a. None of the random variable terms in $\mathcal{F}$ is a finding random variable term.

5b. Each resident random variable term in $\mathcal{F}$ is either a random variable term that consists of a random variable symbol followed by a parenthesized list of one or more ordinary variable symbols, or the random variable symbol $\Diamond$ followed by a constant symbol enclosed in parentheses. (Implementations may treat $\Diamond(\varepsilon)$ and ε as synonyms.)

5c. The only possible values for the identity random variable $\Diamond(\varepsilon)$ are ε and $\perp$. Furthermore, $\Diamond(\mathsf{T})=\mathsf{T}$; $\Diamond(\mathsf{F})=\mathsf{F}$; and $\Diamond(\perp)=\perp$.[8]

5d. For any resident random variable term ψ other than the identity, the local distribution for ψ must assign probability zero to any unique identifier ε for which $\Diamond(\varepsilon) \neq \varepsilon$. One way to ensure this constraint is met is to make $\Diamond(\varepsilon)$ a parent of ψ for any possible value ε for which there is non-zero probability that $\Diamond(\varepsilon) \neq \varepsilon$, and to specify a local distribution that assigns probability zero to ε if $\Diamond(\varepsilon) \neq \varepsilon$. ■

Requirement 5*b* may seem restrictive at first. For example, if we want to assert that the temperature sensor is malfunctioning in the machine most recently inspected by Wilson, we cannot simply define an MFrag with resident random variable *SensorStatus*(*MostRecently-InspectedMachine*(*Wilson*)) and give probability 1 to the value !*Malfunctioning*. The random variable *SensorStatus*(*MostRecentlyInspectedMachine*(*Wilson*)) is defined in its function composition MFrag, and cannot be overridden by the knowledge engineer. We can specify the desired information either by making (*m*=*MostRecentlyInspectedMachine*(*Wilson*)) a parent of *Sensor-Status*(*m*), or by defining a finding random variable Φ(*SensorStatus*(*MostRecentlyInspected-Machine*(*Wilson*))). Which choice is best depends on the circumstances. If Wilson is a saboteur who broke the sensor in the machine he just inspected, the former choice would be appropriate, because whether Wilson inspected the machine has a causal influence on the generative distribution for the machine's sensor status. If there is nothing other than Wilson's recent report of a broken sensor to distinguish Wilson or this machine from any other inspector or machine, then our knowledge is evidential rather than causal, and the latter choice would be appropriate. In either case, the function composition MFrags represent knowledge about the logical properties of function composition. Similar statements apply to the other built-in MFrags. The restrictions on domain-specific MFrags prevent modelers from violating these logical relationships.

[8] A finite domain can be represented by specifying an ordering ε_1, ε_2,… on the unique identifiers, and specifying a probability of 1 that $\Diamond(\varepsilon_{i+1}) = \perp$ if $\Diamond(\varepsilon_i) = \perp$. In this case, the cardinality of the domain is the last i for which $\Diamond(\varepsilon_i) \neq \perp$. The cardinality may of course be uncertain.

In summary, MFrags represent influences among clusters of related random variables. Repeated patterns can be represented using ordinary variables as placeholders into which entity identifiers can be substituted. Probability information for an MFrag's resident random variables are specified via local distributions, which map influence counts for a random variable's parents to probability distributions over its possible values. When ordinary variables appear in a parent but not in a child, the local distribution specifies how to combine influences from multiple copies of the parent random variables. Restricting variable bindings to unique identifiers prevents double counting of repeated instances. Multiple ways of referring to an entity are handled through built-in MFrags that enforce logical constraints on function composition. Built-in logical MFrags give MEBN the expressive power of first-order logic. Context constraints permit recursive relationships to be specified without circular references.

3.3 MEBN Theories

A *MEBN theory* is a collection of MFrags that satisfies consistency constraints ensuring the existence of a unique joint probability distribution over the random variables mentioned in the theory. The built-in MFrags provide logical content and the domain-specific MFrags provide empirical content. This section defines a MEBN theory and states the main existence theorem, that a joint distribution exists for the random variable instances of a MEBN theory. A proof is given in the Appendix.

A MEBN theory containing only generative domain-specific MFrags is called a *generative MEBN theory*. Generative MEBN theories can be used to express domain-specific ontologies that capture statistical regularities in a particular domain of application. MEBN theories with findings can augment statistical information with particular facts germane to a given reasoning problem. MEBN uses Bayesian learning to refine domain-specific ontologies to incorporate observed evidence.

The MFrags of Figure 2 specify a generative MEBN theory for the equipment diagnosis problem. These MFrags specify local probability distributions for their resident random variables. The conditioning constraints in each MFrag specify type restrictions (e.g., the symbol *m* must be replaced by an identifier for an entity of type *Machine*) and functional relationships an influencing configuration must satisfy (e.g., the room identifier *r* must be equal to the value of *MachineLocation*(*m*)). Each local distribution provides a rule for calculating the distribution of a resident random variable given any instance of the MFrag.

Reasoning about a particular task proceeds as follows. First, finding MFrags are added to a generative MEBN theory to represent task-specific information. Next, random variables are identified to represent queries of interest. Finally, Bayesian inference is applied to compute a response to the queries. Bayesian inference can also be applied to refine the local distributions and/or MFrag structures given the task-specific data. For example, to assert that the temperature light is blinking in the machine denoted by *!Machine*37, which is located in the room denoted by *!Room*103*A*, we could add the findings Φ(*TempLight*(*!Machine*37)=*!Blinking*) and Φ(*Machine-Location*(*Machine*37)=*!Room*103*A*) to the generative MEBN theory of Figure 2. To inquire about the likelihood that there are any overheated engines, the FOL sentence $\exists m$ (*Isa*(*Machine*,*m*)$\wedge$(*EngineStatus*(*m*)=*!Overheated*)) would be translated into the quantifier random variable instance $\exists$($m, *Isa*(*Machine*,$m)$\wedge$(*EngineStatus*($m)=*!Overheated*)). A Bayesian inference algorithm would be applied to evaluate its posterior probability given the evidence.

As with ordinary Bayesian networks, global consistency conditions are required to ensure that the local distributions collectively specify a well-defined probability distribution over

interpretations. Specifically, the MFrags must combine in such a way that no random variable instance can directly or indirectly influence itself, and initial conditions must be specified for recursive definitions. Non-circularity is ensured in ordinary Bayesian networks by defining a partial order on random variables and requiring that a random variable's parents precede it in the partial ordering. In dynamic Bayesian networks, random variables are indexed by time, an unconditional distribution is specified at the first time step, and each subsequent distribution may depend on the values of the random variables at the previous time step. Non-circularity is ensured by prohibiting links from future to past and by requiring that links within a time step respect the random variable partial ordering. Other kinds of recursive relationships, such as genetic inheritance, have been discussed in the literature (cf., Pfeffer, 2000). Recursive Bayesian networks (Jaeger, 2001) can represent a very general class of recursively specified probability distributions for logical random variables on finite domains. MEBN provides a very general ability to express recursive relationships on finite or countably infinite domains.

***Definition 6*:** Let $\mathcal{T}' = \{\mathcal{F}_1, \mathcal{F}_2 \ldots\}$ be a set of MFrags. The sequence $\phi_d(\varepsilon_d) \rightarrow \phi_{d-1}(\varepsilon_{d-1}) \rightarrow \ldots \rightarrow \phi_0(\varepsilon_0)$ is called an *ancestor chain* for $\mathcal{T}'$ in partial world state S if there exist $\mathcal{B}_0, \ldots, \mathcal{B}_d$ such that:

6a. Each $\mathcal{B}_i$ is a binding set for one of the MFrags $\mathcal{F}_i \in \mathcal{T}'$;

6b. The random variable instance $\phi_i(\varepsilon_i)$ is obtained by applying the bindings in $\mathcal{B}_i$ to a resident random variable term $\phi_i(\theta_i)$ of $\mathcal{F}_i$;

6c. For $i<d$, either:

- $\phi_{i+1}(\varepsilon_{i+1})$ is obtained by applying the bindings in $\mathcal{B}_i$ to an input random variable term $\phi_{i+1}(\theta_{i+1})$ of $\mathcal{F}_i$, and there is an influencing configuration for $\phi_i(\varepsilon_i)$ and $\mathcal{B}_i$ that contains $\phi_{i+1}(\theta_{i+1})$, or
- $\phi_{i+1}(\varepsilon_{i+1})$ is obtained by applying the bindings in $\mathcal{B}_i$ to a context value assignment term $\phi_{i+1}(\theta_{i+1})$ of $\mathcal{F}_i$.

The integer d is called the *depth* of the ancestor chain. The random variable instance $\phi_j(\varepsilon_j)$ is an *ancestor* of $\phi_0(\varepsilon_0)$ if there exists an ancestor chain $\phi_d(\varepsilon_d) \rightarrow \ldots \rightarrow \phi_j(\varepsilon_j) \rightarrow \ldots \rightarrow \phi_0(\varepsilon_0)$ for $\mathcal{T}'$. ■

***Definition 7*:** Let $\mathcal{T}' = \{\mathcal{F}_1, \mathcal{F}_2 \ldots\}$ be a set of built-in, generative, and/or finding MFrags. Let $\mathcal{V}_{\mathcal{T}'}$ denote the set of random variable terms contained in the $\mathcal{F}_i$, and let $\mathcal{N}_{\mathcal{T}'}$ denote the set of random variable instances that can be formed from $\mathcal{V}_{\mathcal{T}'}$ by substituting the unique identifiers of $\{\mathsf{T}, \mathsf{F}, \perp\} \cup \mathcal{E}$ for arguments of the random variables in $\mathcal{V}_{\mathcal{T}'}$. $\mathcal{T}'$ is a *simple MEBN theory* if the following conditions hold:

7a. No cycles. No random variable instance is an ancestor of itself;[9]

7b. Bounded causal depth. For any random variable instance $\phi(\varepsilon) \in \mathcal{N}_{\mathcal{T}'}$ containing the (possibly empty) unique identifier symbols ε, there exists an integer $N_{\phi(\varepsilon)}$ such that if $\phi_d(\varepsilon_d) \rightarrow \phi_{d-1}(\varepsilon_{d-1}) \rightarrow \ldots \rightarrow \phi(\varepsilon)$ is an ancestor chain for $\mathcal{T}'$ in S, then $d \leq N_{\phi(\varepsilon)}$. The smallest such $N_{\phi(\varepsilon)}$ is called the *depth* $d_{\phi(\varepsilon)}$ of $\phi(\varepsilon)$.

7c. Unique home MFrags. For each $\phi(\varepsilon) \in \mathcal{N}_{\mathcal{T}'}$, there exists exactly one MFrag $\mathcal{F}_{\phi(\varepsilon)} \in \mathcal{T}'$, called the *home MFrag* of $\phi(\varepsilon)$, such that $\phi(\varepsilon)$ is an instance of a resident random variable $\phi(\theta)$ of $\mathcal{F}_{\phi(\varepsilon)}$.[10]

[9] This condition can be relaxed as long as it can be demonstrated that the local distributions are specified non-circularly.

[10] It may be desirable to relax this condition. For example, in an independence of causal influence model, it might be convenient to specify influences due to different clusters of related causes to be specified in separate MFrags. In a

7d. Recursive specification. $\mathcal{T}$ may contain infinitely many domain-specific MFrags, but if so, the MFrag specifications must be recursively enumerable. That is, there must be an algorithm that lists a specification (i.e., an algorithm that generates the input, output, context random variables, fragment graph, and local distributions) for each MFrag in turn, and eventually lists a specification for each MFrag of $\mathcal{T}$. ■

Condition 7*c* simplifies the theoretical analysis, but there are many circumstances in which it would be useful to relax it. For example, in an independence of causal influence model, it might be convenient to specify influences due to different clusters of related causes to be specified in separate MFrags. In a polymorphic version of MEBN, it might be convenient to specify local distributions for separate subtypes in separate MFrags (Costa, 2005). Relaxing Condition 7*c* would also allow a more natural treatment of structural learning. It is clear that the main results of this paper would remain valid under appropriately weakened conditions. Costa (2005) defines a typed version of MEBN that relaxes Condition 7*c*.

***Theorem 1*:** Let $\mathcal{T} = \{ \mathcal{F}_1, \mathcal{F}_2 \ldots \}$ be a simple MEBN theory. There exists a joint unique probability distribution $\mathcal{P}^{\text{gen}}_{\mathcal{T}}$ on the set of instances of the random variables of its MFrags that is consistent with the local distributions assigned by the MFrags of $\mathcal{T}$. This distribution respects the independence assumptions encoded in the MFrags. That is, a random variable instance is conditionally independent of its non-descendants given its full (possibly infinite) set of parent instances. ■

The proof of Theorem 1 is found in the appendix.

MEBN inference conditions the joint probability distribution implied by Theorem 1 on the proposition that all findings have value T. This conditional distribution clearly exists if there is a non-zero probability that all findings have value T. However, when there is an infinite sequence of findings or there are findings on quantifier random variables, then any individual sequence of findings may have probability zero even though some such sequence is certain to occur. For example, each possible realization of an infinite sequence of rolls of a fair die has zero probability, yet some such sequence will occur if tossing continues indefinitely. Although any individual sequence of tosses has probability zero, the assumption that the die is fair allows us to draw conclusions about properties of the sequences of tosses that will actually occur. In particular, it is a practical (although not a logical) certainty that if the die is fair, then the limiting frequency of rolling a four will be once in every six trials. That is, although a sequence having limiting probability 1/6 and a sequence having limiting probability 1/3 both have probability zero, the set of worlds in which the limit is 1/6 is infinitely more probable than the set of worlds in which the limit is 1/3. Practical certainties about stochastic phenomena are formalized as propositions that are true "almost surely" or "except on a set of measure zero" (Billingsley, 1995). Almost sure propositions are not true in all possible interpretations of the FOL theory corresponding to a MEBN theory, but the set of worlds in which they are true has probability 1 under the probability distribution represented by the MEBN theory. In the above example, the set of worlds in which the limiting frequency is1/6 has probability 1.

The following results pertain to the existence of conditional distributions in a MEBN theory.

---

polymorphic version of MEBN logic, it might be convenient to specify local distributions for separate subtypes in separate MFrags. It is clear that the main results would remain valid under appropriately weakened conditions.

***Definition 8*:** The distribution $P_{\mathcal{T}'}^{\text{gen}}$ is called the *generative or prior distribution* for $\mathcal{T}'$. Let $\{\Phi(\psi_1=\alpha_1), \Phi(\psi_2=\alpha_2), \ldots\}$ be the finding MFrags for $\mathcal{T}'$. A *finding alternative* for $\mathcal{T}'$ is a set $\{\Phi(\psi_1=\alpha'_1), \Phi(\psi_2=\alpha'_2), \ldots\}$ of values for the finding random variables of $\mathcal{T}'$, possibly assigning different values to the finding random variables from the values assigned by $\mathcal{T}'$. Finding alternatives represent counterfactual worlds for $\mathcal{T}'$ – that is, worlds that were *a priori* possible but are different from the world asserted by the findings to have occurred. ■

***Corollary 2*:** Let $\mathcal{T}'$ be a MEBN theory with findings $\{\Phi(\psi_1=\alpha_1), \Phi(\psi_2=\alpha_2), \ldots\}$. Then a conditional distribution exists for $P_{\mathcal{T}'}^{\text{gen}}$ given $\{\psi_1, \psi_2, \ldots\}$. Furthermore, any two such distributions differ at most on a set of finding alternatives assigned probability zero by $P_{\mathcal{T}'}^{\text{gen}}$. The same holds for a conditional distribution given any finite-length subsequence $\{\psi_1, \psi_2, \ldots, \psi_n\}$. ■

Corollary 2 follows immediately from Theorem 1 and the Radon-Nikodym Theorem (Billingsley, 1995). A distribution $P_{\mathcal{T}'}(\xi_1, \xi_2 \ldots \mid \Phi(\psi_1 = \alpha_1), \Phi(\psi_2 = \alpha_2), \ldots)$ for a set of random variables $\{\xi_1, \xi_2, \ldots\}$ obtained by conditioning $P_{\mathcal{T}'}^{\text{gen}}$ on all findings having value T is abbreviated $P_{\mathcal{T}'}(\xi \mid \Phi(\psi = \alpha))$, and is called a posterior distribution for ξ given the findings $\Phi(\psi=\alpha)$. Any two posterior probabilities are equal except on a set of finding alternatives assigned probability zero by $P_{\mathcal{T}'}^{\text{gen}}$.

When the sequence of finding random variables is infinite, the probability of any particular realization $(\psi=\alpha)$ may be zero. In this case, there is no single well-defined conditional distribution for ξ given $(\psi=\alpha)$. In fact, the conditional distribution given $(\psi=\alpha)$ can be set arbitrarily to any distribution whatsoever. This would seem to imply that Theorem 2 is vacuous. However, much of statistical theory and practice concerns conditioning on probability zero events. In many important problems, one particular conditional distribution is singled out as the most natural one, as the limiting distribution of an appropriate sequence of distributions conditioned on events with non-zero probability (cf., DeGroot and Schervish, 2002, p. 105). Theorem 3 implies that this is the case for most sequences, in the sense that $P_{\mathcal{T}'}^{\text{gen}}$ assigns probability 1 to sequences for which there is a well-defined limiting distribution.

***Theorem 3*:** Suppose $\{(\xi_1=\gamma_1), (\xi_2=\gamma_2), \ldots, (\xi_n=\gamma_n)\}$, abbreviated $(\xi=\gamma)$, is an assignment of values to a finite set of random variables of $\mathcal{T}'$. Then $P_{\mathcal{T}'}^{\text{gen}}$ assigns probability 1 to the set of finding alternatives $\{\Phi(\psi_1=\alpha'_1), \Phi(\psi_2=\alpha'_2), \ldots\}$ for which

$$\lim_{n\to\infty}\left(P_{\mathcal{T}'}((\xi = \gamma) \mid \Phi(\psi_1 = \alpha'_1), \Phi(\psi_2 = \alpha'_2), \ldots, \Phi(\psi_n = \alpha'_n))\right)$$

exists and is equal to $P_{\mathcal{T}'}((\xi = \gamma) \mid \Phi(\psi = \alpha'))$. ■

To prove Theorem 3, consider the sequence $X_1, X_2, \ldots$ of random variables defined by:

$$X_n = P_{\mathcal{T}'}((\xi = \gamma) \mid \psi_1, \psi_2, \ldots, \psi_n).$$

Let $\mathcal{F}_n$ be the σ-field generated by the first n finding random variables $\psi_1, \psi_2, \ldots, \psi_n$. The random variables $X_1, X_2, \ldots$ satisfy the condition that $E[X_{n+1} \mid \mathcal{F}_n] = X_n$. A sequence that satisfies this property with respect to a σ-field is called a martingale with respect to that σ-field. The X_n also satisfy $\sup_n E[\,|X_n|\,] < \infty$. The martingale convergence theorem (Billingsley, 1995) implies that there is a random variable X such that $X_n \to X$ with probability 1. Furthermore, Theorem 35.5 of Billingsley (1995) implies that $P_{\mathcal{T}'}^{\text{gen}}$ assigns probability 1 to the event that $X = P_{\mathcal{T}'}((\xi = \gamma) \mid \psi_1, \psi_2, \ldots)$.

4 Semantics, Representation Power, and Inference

Section 4.1 defines model theoretic semantics of MEBN. Section 4.2 demonstrates that multi-entity Bayesian networks as formalized in Section 3 can express a probability distribution over interpretations of any classical first-order theory, and constructs a MEBN theory in which every satisfiable sentence has non-zero probability. Section 4.3 describes an algorithm for performing inference with MEBN theories.

4.1 MEBN Semantics

In the standard model theoretic semantics for first-order logic developed by Tarski (1944; c.f., Enderton, 2001), a FOL theory is interpreted in a domain by assigning each constant symbol to an element of the domain, each function symbol on k arguments to a function mapping k-tuples of domain elements to domain elements, and each predicate symbol on k arguments to a subset of k-tuples of domain elements corresponding to the entities for which the predicate is true (or, equivalently, to a function mapping k-tuples of domain elements to truth-values). If the axioms are consistent, then there exists a domain and an interpretation such that all the axioms of the theory are true assertions about the domain, given the correspondences defined by the interpretation. Such an interpretation is called a model for the axioms.

MEBN theories define probability distributions over interpretations of an associated FOL theory. Each k-argument random variable in a MEBN theory represents a function mapping k-tuples of unique identifiers to possible values of the random variable. Any function consistent with the logical constraints of the MEBN theory is allowable, and the probability that the function takes on given values is specified by the joint probability distribution represented by the MEBN theory. For logical random variables, the possible values of the function are T, F, and $\bot$; for phenomenal random variables, the possible values are entity identifiers and $\bot$. Through the correspondence between entity identifiers and entities in the domain, a random variable also represents a function mapping k-tuples of domain entities either to domain entities (for phenomenal random variables) or to truth-values of assertions about the domain (for logical random variables).

MEBN provides a logically coherent means of specifying a global joint distribution by composing local conditional distributions involving small sets of random variables. Formerly, this could be achieved only for restricted kinds of distributions. Standard Bayesian networks allow joint distributions on a finite number of random variables to be composed from locally defined conditional distributions. There are well-known special cases, such as independent and identically distributed sequences or Markov chains, for which joint distributions on infinite sets of random variables can be composed from locally defined conditional distributions. MEBN provides the ability to construct joint distributions from local elements for a much wider class of distributions on infinite collections of random variables. As demonstrated below, MEBN can represent a joint distribution over first-order sentences that assigns non-zero probability to every satisfiable sentence. Thus, through Bayesian conditioning, a probability distribution can be expressed on interpretations of any consistent, finitely axiomatizable first-order theory. This distribution can be updated through Bayesian conditioning when new axioms are added, providing a theoretical framework for analyzing limiting distributions over interpretations of infinite sequences of first-order sentences.

Consider a MEBN theory $\mathcal{T}_M$ in a language $\mathcal{L}_M$ having phenomenal random variable symbols $\mathcal{X}=\{\xi_i\}$, phenomenal constant symbols $\mathcal{A}=\{\alpha_i\}$, domain-specific logical random variable symbols $\mathcal{B}=\{\beta_i\}$, exemplar symbols $S=\{\sigma_{\phi i}\}$ and entity identifier symbols $\mathcal{E}=\{\varepsilon_i\}$. It is assumed that the

sets $\mathcal{X}$, $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{E}$ are pairwise disjoint, are either finite or countably infinite, and do not contain the symbols T, F, or $\perp$. It is assumed that S contains a distinct exemplar symbol $\sigma_{\phi i} \notin \mathcal{X} \cup \mathcal{A} \cup \mathcal{B} \cup \mathcal{E} \cup \{\mathsf{T},\mathsf{F},\perp\}$ for each pair consisting of an open logical random variable term $\phi(\gamma_1,\ldots,\gamma_n)$ of $\mathcal{L}_M$ and index i of an ordinary variable γ_i occurring in $\phi(\gamma_1,\ldots,\gamma_n)$.

The following conditions will be assumed in this section because they make it straightforward to define a correspondence between a MEBN theory and a counterpart FOL theory with the same logical content. These conditions are not requirements of MEBN, and need not be assumed by any given application. Even when they are not satisfied, a MEBN theory defines a probability distribution on interpretations of a first-order theory, but defining the correspondence is less straightforward.

- FOL1: There are no quantifier random variable terms among the context terms in any of the MFrags of $\mathcal{T}_M$, and no simple random variable term of $\mathcal{T}_M$ has a quantifier random variable term as a parent.
- FOL2: Random variables $\xi \in \mathcal{X}$ or $\beta \in \mathcal{B}$ have value $\perp$ if any of their arguments belong to $\{\mathsf{T}, \mathsf{F}, \perp\}$;
- FOL3: If the values of all arguments to a phenomenal random variable ξ belong to $\mathcal{E}$, then the value of ξ belongs to $\mathcal{E}$ with probability 1;
- FOL4: Any constant symbol $\alpha \in \mathcal{A}$ has value in $\mathcal{E}$ with probability 1;
- FOL5: If the values of all arguments to a logical random variable β belong to $\mathcal{E}$, then the value of β belongs to $\{\mathsf{T}, \mathsf{F}\}$ with probability 1.

Given these conditions, $\mathcal{P}^{\text{gen}}_{\mathcal{T}_M}$ generates random interpretations of the phenomenal random variable symbols of $\mathcal{L}_M$ in the domain $\{\varepsilon \in \mathcal{E} : \Diamond(\varepsilon) \neq \perp)\}$ of meaningful entity identifiers. That is, for each constant symbol, $\mathcal{P}^{\text{gen}}_{\mathcal{T}_M}$ generates a meaningful entity identifier. For each phenomenal random variable symbol, $\mathcal{P}^{\text{gen}}_{\mathcal{T}_M}$ generates a random function mapping k-tuples of meaningful entity identifiers to meaningful entity identifiers. For each logical random variable symbol, $\mathcal{P}^{\text{gen}}_{\mathcal{T}_M}$ generates a random function mapping k-tuples of meaningful entity identifiers to $\{\mathsf{T}, \mathsf{F}\}$ (or equivalently, the subset of k-tuples for which the randomly generated function has value T).

A classical first-order theory $\mathcal{T}_F$ that represents the logical content of $\mathcal{T}_M$ is defined as follows:

1. The language $\mathcal{L}_F$ for $\mathcal{T}_F$ has function symbols $\mathcal{X}$, constant symbols $\mathcal{A} \cup \mathcal{E} \cup \{\perp\}$, and predicate symbols $\mathcal{B}$, where the number of arguments for functions and predicates in $\mathcal{L}_F$ is the same as the number of arguments for the corresponding random variables in $\mathcal{T}_M$.
2. For each pair ε_1 and ε_2 of distinct entity identifiers, $\mathcal{T}_F$ contains an axiom $(\varepsilon_1 = \varepsilon_2) \Rightarrow (\varepsilon_1 = \perp) \wedge (\varepsilon_2 = \perp)$.
3. For each phenomenal random variable symbol ξ, $\mathcal{T}_F$ contains axioms asserting that no instance of ξ may take on values outside the set of possible values as defined in the home MFrag for ξ.
4. If a local distribution in a domain-specific MFrag of $\mathcal{T}_M$ assigns probability zero to possible value ε of a phenomenal resident random variable $\xi(x)$ for some set $\#S_{W\xi(x)}$ of influence counts, there is an axiom of $\mathcal{T}_F$ specifying that the function corresponding to $\xi(x)$ is not equal to ε when the context constraints hold and the parents of $\xi(x)$ satisfy $\#S_{W\xi(x)}$. Each such axiom is universally quantified over any ordinary variables appearing in ξ and/or its parents and/or the context random variables in the home MFrag of ξ. Formally, $\mathcal{T}_F$ contains an axiom $\forall x\, ((\kappa(x) \wedge \#S_{W\xi(x)}) \Rightarrow \neg(\xi(x) = \varepsilon))$. Here, $\kappa(x)$ and $\#S_{W\xi(x)}$ denote formulae in $\mathcal{L}_F$ asserting that the context constraints hold and that the influence

counts for the parents of $\xi(x)$ are equal to $\#S_{W\xi(x)}$; and x denotes any ordinary variables on which ξ, κ, and/or the parents of ξ depend. This applies also to constant random variables, which are treated as functions with no arguments.

5. If a local distribution in a domain-specific MFrag of $\mathcal{T}'_M$ assigns probability one to T for a logical random variable $\beta(x)$ for some set $\#S_{W\beta(x)}$ of influence counts, there is an axiom of $\mathcal{T}'_F$ specifying that the predicate $\beta(x)$ is true under these conditions. That is, $\mathcal{T}'_F$ contains an axiom $\forall x\ ((\kappa(x) \wedge \#S_{W\beta(x)}) \Rightarrow \beta(x))$. Here, $\kappa(x)$ and $\#S_{W\beta(x)}$ denote formulae in $\mathcal{L}_{\mathcal{T}}$ asserting that the context constraints hold and that the influence counts for the parents of $\beta(x)$ are equal to $\#S_{W\beta(x)}$, respectively; and x denotes any ordinary variables on which β, κ, and/or the parents of β depend.
6. If a local distribution in a domain-specific MFrag of $\mathcal{T}'_M$ assigns probability one to F for a logical random variable $\beta(x)$ for some set $\#S_{W\beta(x)}$ of influence counts, there is an axiom of $\mathcal{T}'_F$ specifying that the predicate $\beta(x)$ is false under these conditions. That is, $\mathcal{T}'_F$ contains an axiom $\forall x\ ((\kappa(x) \wedge \#S_{W\beta(x)}) \Rightarrow \neg\beta(x))$. Here, $\kappa(x)$ and $\#S_{W\beta(x)}$ denote formulae in $\mathcal{L}_{\mathcal{T}}$ asserting that the context constraints hold and that the influence counts for the parents of $\beta(x)$ are equal to $\#S_{W\beta(x)}$, respectively; and x denotes any ordinary variables on which β, κ, and/or the parents of β depend.

The logical combination MFrags (see Figure 8) define random variables that explicitly represent the truth-values of sentences of $\mathcal{T}'_F$. The assumptions FOL1-FOL5 ensure that these truth-values satisfy the axioms defining $\mathcal{T}'_F$. That is, $\mathcal{P}^{\text{gen}}_{\mathcal{T}'_M}$ generates random models of the axioms of $\mathcal{T}'_F$. However, there may be sentences satisfiable under the axioms of $\mathcal{T}'_F$ to which $\mathcal{P}^{\text{gen}}_{\mathcal{T}'_M}$ assigns probability zero. When a satisfiable sentence of $\mathcal{T}'_F$ is assigned probability zero by $\mathcal{P}^{\text{gen}}_{\mathcal{T}'_M}$, there is no assurance that a well-defined conditional distribution exists given that the corresponding logical random variable has value T. The following additional condition ensures that a well-defined conditional distribution exists given any finite set of satisfiable findings on random variables of $\mathcal{T}'_M$. Again, this assumption is not required by MEBN, and may not be appropriate for a given application.

FOL6: Suppose $\phi(\gamma_1,\ldots,\gamma_n)$ is an open logical random variable of $\mathcal{T}'_M$ that depends on ordinary variables $\gamma_1,\ldots,\gamma_n$. Let $\theta_1(\sigma_{\phi 1}, \theta_2(\sigma_{\phi 2}, \ldots, \theta_n(\sigma_{\phi n}, \phi(\sigma_{\phi 1}, \sigma_{\phi 2}, \ldots, \sigma_{\phi n}))))$ be a quantifier random variable, where $\sigma_{\phi i}$ is the exemplar symbol for ordinary variable γ_i and θ_i stands for one of the quantifier symbols $\exists$ or $\forall$. Suppose this quantifier random variable $\theta_1(\sigma_{\phi 1}, \theta_2(\sigma_{\phi 2}, \ldots, \theta_n(\sigma_{\phi n}, \phi(\sigma_{\phi 1}, \sigma_{\phi 2}, \ldots, \sigma_{\phi n}))))$ corresponds to a satisfiable formula of $\mathcal{T}'_F$. Then $\mathcal{P}^{\text{gen}}_{\mathcal{T}'_M}$ assigns strictly positive probability to the value T for the quantifier random variable $\theta_1(\sigma_{\phi 1}, \theta_2(\sigma_{\phi 2}, \ldots, \theta_n(\sigma_{\phi n}, \phi(\sigma_{\phi 1}, \sigma_{\phi 2}, \ldots, \sigma_{\phi n}))))$.

***Corollary 4*:** Suppose $\mathcal{T}'_M$ satisfies FOL1-FOL6, and suppose that $\mathcal{T}'_F$ is the first-order theory, constructed as above, expressing the logical content of $\mathcal{T}'_M$. Let $\{\Phi(\psi_1=\alpha_1), \Phi(\psi_2=\alpha_2), \ldots, \Phi(\psi_n=\alpha_n)\}$ be a finite set of findings such that the conjunction of the $(\psi_i=\alpha_i)$ is satisfiable as a sentence of $\mathcal{T}'_F$. Let $\{\alpha, \xi(\theta), \beta(\theta)\}$ stand for the set of all instances of constant, phenomenal, and logical random variables of $\mathcal{T}'_F$. Then the posterior distribution $\mathcal{P}_{\mathcal{T}'_M}\left(\{\alpha, \xi(\theta), \beta(\theta)\} \mid \Phi(\psi = \alpha)\right)$ exists and is unique. ■

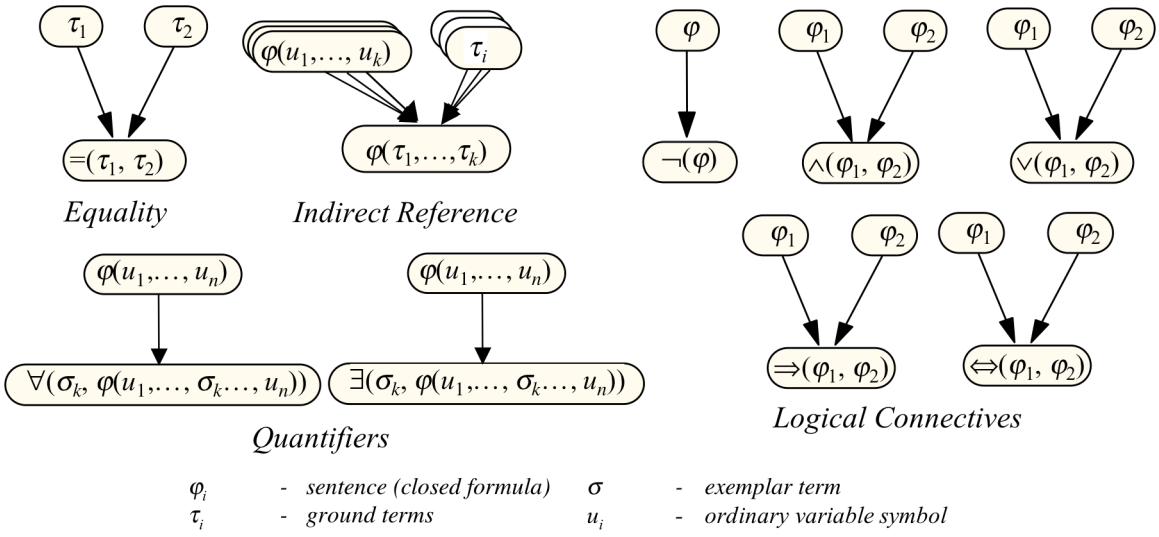


Figure 8: Logical MFrags

Corollary 4 is a straightforward consequence of Corollary 2. Specifying a generative distribution that satisfies FOL1-FOL5 is relatively straightforward. A construction is provided in Section 4.2 of a MEBN theory $\mathcal{T}_{M^*}$ for which $\mathcal{P}^{\text{gen}}_{\mathcal{T}_{M^*}}$ satisfies FOL6.

A MEBN theory is interpreted in a domain of application by associating each entity identifier symbol with an entity in the domain. Through this correspondence between identifiers and the entities they represent, a MEBN theory induces a probability distribution on attributes of and relationships among entities in the domain of application. In particular, although the generative distribution for a MEBN theory constructs interpretations in the countable domain of entity identifiers, a MEBN theory can be applied to reason about domains of any cardinality. Under the assumption that the entities associated with the entity identifiers constitute a representative sample of entities in the domain, statistical conclusions drawn about the domain are valid for domains of any cardinality.

Important advantages of MEBN random variable semantics are clarity and modularity. For example, we could add a new collection of MFrags to our equipment diagnosis MEBN theory, say for reasoning about the vacation and holiday schedule of maintenance technicians, without affecting the probabilities of any assertions unrelated to the change. Furthermore, the probability distribution represented by a MEBN theory is a well-defined mathematical object independent of its correspondence with actual objects in the world, having a clearly specified semantics as a probability distribution on interpretations. Its adequacy for reasoning about the actual world rests in how well the relationships in the model reflect the empirical relationships among the entities to which the symbols refer in a given domain of application. Our approach thus enforces a distinction between logical and empirical aspects of a representation and provides a clearly defined interface between the two. This supports a principled approach to empirical evaluation and refinement of domain ontologies.

4.2 A Generative Distribution for First-Order Logic

This section demonstrates how to define a generative MEBN theory $\mathcal{T}_{M^*}$ such that $\mathcal{P}^{\text{gen}}_{\mathcal{T}_{M^*}}$ places positive probability on value T for any logical random variable ϕ that corresponds to a satisfiable sentence in first-order logic.

Consider a MEBN language $\mathcal{L}_{M^*}$ and classical FOL language $\mathcal{L}_{F^*}$ related to each other as described in Section 4.1. We assume there is a total ordering φ_1, φ_2, … of the phenomenal constant, phenomenal and logical random variable terms $\varphi_i \in \mathcal{A} \cup \mathcal{X} \cup \mathcal{B}$, and a total ordering ε_1, ε_2, …$\in \mathcal{E}$ of entity identifiers. The domain-specific MFrags of a generative MEBN theory must define a distribution for each simple open random variable term $\varphi_i(u_1,\ldots,u_{n_i})$, where the u_j are ordinary variables and n_i is the number of arguments taken by φ_i. A distribution is also defined for the exemplar constants. The remaining random variables are defined via the logical MFrags of Figure 8.

The joint distribution for simple open random variables and exemplar constants is defined as follows. Let ψ_1, ψ_2, … be a total ordering of the quantifier random variables; let π_1, π_2, … be a strictly positive probability distribution on the entity identifiers, and let $0 < \theta, \rho < 1$ be real numbers. We use the notation ψ_k to refer a quantifier random variable and σ_{ψ_k} to refer to the exemplar constant for ψ_k. That is, ψ_k denotes a logical random variable of the form $\forall(\sigma_{\psi_k}, \phi(\sigma_{\psi_k}))$ or $\exists(\sigma_{\psi_k}, \phi(\sigma_{\psi_k}))$, where $\phi(u)$ is an open logical random variable called the *body* of ψ_k. We can think of the exemplar constant σ_{ψ_k} as denoting a generic filler entity for its place in the quantifier random variable. Its generative distribution is defined in a way that ensures that if the negation of its quantifier random variable is satisfiable, then σ_{ψ_k} has a non-zero probability of having the value $\perp$, which constrains the corresponding quantifier random variable to have value F.

Exemplar constant distributions: The distributions for exemplar constants are defined inductively such that the exemplar term $\Diamond(\sigma_{\psi_k})$ has value $\perp$ in models in which ψ_k is constrained to have value F, and otherwise is sampled randomly from the entity identifiers that are logically possible values for σ_{ψ_k}. Specifically:

- The parents of $\Diamond(\sigma_{\psi_k})$ are $\Diamond(\sigma_{\psi_1}), \Diamond(\sigma_{\psi_2}), \ldots$, and $\Diamond(\sigma_{\psi_{k-1}})$.
- By the inductive hypothesis, it is assumed that if $\Diamond(\sigma_{\psi_i}) = \perp$ for $i < k$, then ψ_k has value F. (It will be verified below that the inductive hypothesis is true for k, then it is true for $k + 1$.) Conditional on $\Diamond(\sigma_{\psi_1}), \Diamond(\sigma_{\psi_2}), \ldots$, and $\Diamond(\sigma_{\psi_{k-1}})$, the distribution of $\Diamond(\sigma_{\psi_k})$ is defined as follows:
  - If ψ_k is unsatisfiable as a formula of $\mathcal{L}_{F^*}$ given the constraints on ψ_1, …, ψ_{k-1} implied by the values of its parents, then $\Diamond(\sigma_{\psi_k})$ has value $\perp$ with probability 1.
  - If $\neg\psi_k$ is unsatisfiable as a formula of $\mathcal{L}_{F^*}$ given the constraints on ψ_1, …, ψ_{k-1} implied by the values of its parents, then $\Diamond(\sigma_{\psi_i})$ has value ε_j with probability π_j.
  - Otherwise, $\Diamond(\sigma_{\psi_i})$ has value $\perp$ with probability θ and ε_j with probability $(1-\theta)\pi_j$.

  Calculating the above probabilities requires checking for satisfiability of ψ_k and $\neg\psi_k$, which is in general undecidable. We can define a process that satisfies Definition 3 as follows. First, we assign probability θ to $\perp$ and $(1-\theta)\pi_j$ to ε_j. Then we execute the satisfiability checker. If at any point ψ_k is proven unsatisfiable, we change the distribution to assign probability 1 to $\perp$. If $\neg\psi_k$ is proven unsatisfiable, we assign probability zero to $\perp$ and π_j to ε_j. If either ψ_k or $\neg\psi_k$ is unsatisfiable, this algorithm will eventually halt with the correct result. Otherwise, it was initialized with the correct distribution and this distribution never changes, so if the algorithm is interrupted it will give the correct result.

Domain-specific random variable distributions: The distribution of $\varphi_k(u_1,\ldots,u_{n_k})$ is defined as follows.

- The parents of $\varphi_k(u_1,\ldots,u_{n_k})$ are:
  - $\varphi_i(v_1,\ldots,v_{n_i})$ for all $i<k$, where v_j is a different ordinary variable than u_j, implying that all instances of $\varphi_i(v_1,\ldots,v_{n_i})$ are parents of each instance of $\varphi_k(u_1,\ldots,u_{n_k})$;
  - Instances of $\varphi_k(v_1,\ldots,v_{n_k})$ such that the entity identifier bound to each u_j is equal to or precedes the entity identifier bound to v_j, and strictly precedes it for at least one j. (This can be specified by a recursive definition with appropriate context constraints);
  - The exemplar constants $\Diamond(\sigma_{\psi_1}), \Diamond(\sigma_{\psi_2}), \ldots$
  - The identity random variables $\Diamond(e)$.
- If $\varphi_k(u_1,\ldots,u_{n_k})$ is a phenomenal random variable, its probability distribution is calculated as follows. For any binding $\varepsilon_1,\ldots,\varepsilon_{n_k}$ of entity identifiers to the variables $u_1,\ldots,u_{n_k}$, the value $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})=\varepsilon_j$ is assigned randomly, with probability π_j, from among the entity identifiers whose value is consistent with the satisfiability constraints implied by the assignment of values to the parents of $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})$. Again, this step requires satisfiability checks. Definition 3 is satisfied if it is implemented by initially assigning probability π_j to ε_j, and if $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})=\varepsilon_j$ is proven unsatisfiable, setting the probability of ε_j to zero. The probability assigned to $\perp$ converges to the correct value, but may never stop changing. This is allowed by Definition 3.
- If $\varphi_k(u_1,\ldots,u_{n_k})$ is a logical random variable, its probability distribution is calculated as follows. For any binding $\varepsilon_1,\ldots,\varepsilon_{n_k}$ of entity identifiers to the variables $u_1,\ldots,u_{n_k}$:
  - $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})$ has value T if $\neg\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})$ is inconsistent with the satisfiability constraints implied by the assignment of values to the parents of $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})$;
  - $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})$ has value F if $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})$ is inconsistent with the satisfiability constraints implied by the assignment of values to the parents of $\varphi_k(\varepsilon_1,\ldots,\varepsilon_{n_k})$;
  - Otherwise, $\varphi_k(u_1,\ldots,u_{n_k})$ has value T with probability ρ and F with probability $(1-\rho)$.

  As before, this calculation is implemented by initially assigning probability ρ to T and probability $(1-\rho)$ to F, and revising the distribution if one of the satisfiability checks fails.

Theorem 5: If ψ is a closed logical random variable corresponding to a satisfiable sentence of $\mathcal{L}_{F^*}$, then $\mathcal{P}^{\text{gen}}_{\mathcal{T}'_{M^*}}$ places non-zero probability on the value T for ψ.

Proof: The above definition ensures that if ψ corresponds to a satisfiable sentence of $\mathcal{T}'_{F^*}$, then there is a non-zero probability that $\Diamond(\sigma_{\neg\psi})$ has value $\perp$. When $\Diamond(\sigma_{\neg\psi})$ has value $\perp$, the local distributions for the domain-specific random variables are assigned in a way that constrains ψ to have value T. Therefore, there is a non-zero probability that ψ has value T. ■

Theorem 5 shows that $\mathcal{P}^{\text{gen}}_{\mathcal{T}'_{M^*}}$ places non-zero probability on the value T for sentences of $\mathcal{L}_{F^*}$ that are consistent with the axioms of $\mathcal{T}'_{F^*}$, which is a first-order theory obtained from $\mathcal{P}^{\text{gen}}_{\mathcal{T}'_{M^*}}$ by following the rules of Section 4.1. The final step in our argument is to show how to use Theorem

5 to define a probability distribution that places non-zero probability on the models of any satisfiable sentence in first-order logic.

Let $\mathcal{L}$ be a first-order language with function symbols $\mathcal{X}$, constant symbols $\mathcal{A}$, and predicate symbols $\mathcal{B}$, and let ψ be a sentence of $\mathcal{L}$. The correspondences defined in the logical MFrags of Figure 8 provide a recipe for defining a language $\mathcal{L}_{M^*}$ that augments $\mathcal{L}$ with the special logical constants and random variables common to all MEBN theories. Following the above definitions, we can define a MEBN theory $\mathcal{T}_{M^*}$ that has the same domain-specific random variable symbols as $\mathcal{L}$. This MEBN theory has a logical random variable ψ_{M^*} that makes the same assertion as ψ. Section 4.1 defines a corresponding sentence ψ_{F^*} of $\mathcal{T}_{F^*}$. By examining how ψ_{M^*} is constructed from ψ and how $\mathcal{T}_{F^*}$ is constructed from $\mathcal{T}_{M^*}$, it is clear that ψ_{F^*} is satisfiable as a sentence of $\mathcal{L}_{F^*}$ if and only if ψ is satisfiable as a sentence of $\mathcal{L}$. Thus, given a first-order language with countably many symbols, we can define a MEBN theory that represents a joint probability distribution over logical random variables corresponding to sentences of $\mathcal{L}$. This MEBN theory assigns non-zero probability to the value T for any logical random variable corresponding to a satisfiable sentence ψ of $\mathcal{L}$. Conditioning on the value T for this random variable results in a joint distribution on models of ψ. Furthermore, the same holds for any finite set of jointly satisfiable sentences, because their conjunction is a satisfiable sentence. It is also clear that this approach fails for infinite sequences of sentences.

4.3 Comments on the FOL Generative Distribution

Some comments are in order to clarify what has been accomplished in defining the probability distribution of Section 4.2. One need not go to such trouble merely to demonstrate the existence of a distribution on interpretations that assigns a positive probability to each sentence. This can be accomplished simply by: (1) listing the satisfiable sentences of the language; (2) invoking the axiom of choice to pick an interpretation to satisfy each sentence in the list; and (3) assigning a probability to each of the chosen interpretations. Of course, this is purely a formal demonstration of existence, and it is not clear whether the resulting distribution has any meaning with respect to the intended domain of application.

The distribution defined in Section 4.2 generates possible worlds by assigning random values to functions and predicates. Because the sampled worlds are required to respect satisfiability constraints that may be undecidable, the generative probabilities are not Turing computable. However, they are computable by an oracle machine with a satisfiability-checking oracle. Oracle machines are often used to prove properties of algorithms. Although the distribution of Section 4.2 cannot be computed exactly, it can be approximated.

Furthermore, it is not difficult to modify the process of Section 4.2 to obtain a generative distribution that is nearly the same as a distribution specified by a domain modeler, but with a "pinch of probability" (DeGroot, 1982) allocated to each satisfiable sentence. This is accomplished by making two changes to the process defined in Section 4.2. First, the exemplar sampling distribution is modified to make probability of imposing an unsatisfiability constraint very small. Second, the domain-specific random variable distributions are sampled according to the domain theory when there are no applicable unsatisfiability constraints, and to localize the changes when there are constraints.

The process of Section 4.2 imposes unsatisfiability constraints by assigning the value $\perp$ to exemplar constants. When the sampler is about to generate the k^{th} exemplar, it first invokes the oracle to check whether either ψ_k or $\neg\ \psi_k$ is unsatisfiable given previously generated constraints. If ψ_k is unsatisfiable, the value $\perp$ is sampled; if $\neg\psi_k$ is unsatisfiable, a non-$\perp$ value is sampled. If

both ψ_k and $\neg\psi_k$ are satisfiable, a random choice is made of whether to impose an unsatisfiability constraint (assign the value $\perp$) or not to impose a constraint (assign a non-$\perp$ value). At this point, another modification is made to the sampling distribution as follows. An integer $k \geq 0$ is chosen with probability $(1\text{-}\theta)k\theta$. If $k \geq 1$ and ψ_k is satisfiable, the sampler sets σ_ψ to the value $\perp$, which will constrain sampling to prevent ψ_k from being satisfied. Otherwise, set σ_ψ is set to the value $\perp$ if and only if ψ_k is unsatisfiable given the constraints imposed thus far, if any. This procedure imposes at most one unsatisfiability constraint, and the probability of imposing a constraint is less than θ.

Next, the sampling distributions for the domain-specific random variables are modified to make the perturbed distribution as close as possible to the original MEBN theory. To do this, sampling of the domain-specific random variables is carried out in an order consistent with the fragment graphs of the original MEBN theory. When sampling random variable φ, if we have not imposed a constraint in our exemplar sampling, or if φ is not an ancestor of a quantifier random variable ψ for which an unsatisfiability constraint was imposed, then φ is sampled according to its local distribution in the original MEBN theory. Otherwise, a distribution is chosen that prevents ψ from being satisfied. To do this, we may need to add new arcs to φ from random variables that are not its parents in the original MEBN theory; but to minimize changes to the original distribution, we do not add parents unless they are ancestors of ψ.

The result of this oracle machine construction is a generative distribution that is close to the original distribution, and approaches it in the limit as θ tends to zero. The unsatisfiability constraints are imposed as local perturbations that disturb the original MEBN theory as little as possible. Specifically, when intervening to make ψ unsatisfiable, we modify only the distributions of ancestors of ψ. The generative distributions of all other random variables remain unchanged.

There are a few subtle issues that deserve mention. One issue is accounting for the possibility that the original MEBN theory itself specifies unsatisfiability constraints. This is possible, but details are not presented here. Second, although the probabilities assigned by this "pinch of probability" distribution can be made as close as we like to the original distribution, there remains an arbitrariness to conditional probabilities given sentences assigned probability zero by the original MEBN theory. The arbitrariness is localized to the generative distributions of ancestors of the measure zero sentence. This contrasts with standard probability theory, in which conditional distribution given a set of measure zero is totally arbitrary. Basing a probabilistic logic on graphical models allows us to localize the effects of conditioning on an event of measure zero. If the original MEBN theory imposes additional mathematical structure, such as a continuous probability density, then there is a natural choice for the conditional distribution. Further analysis of this topic is beyond the scope of this paper.

4.4 MEBN Inference: Situation-Specific Bayesian Networks

As noted above, MEBN inference conditions the prior distribution represented by a MEBN theory on its findings. Figure 9 sketches an inference algorithm that uses knowledge-based model construction (Wellman, et al., 1992) to produce a sequence of *approximate situation-specific Bayesian networks*. Mahoney and Laskey (1998) define a situation-specific Bayesian network (SSBN) as a minimal Bayesian network sufficient to compute the response to a query, where a query consists of obtaining the posterior distribution for a set of target random variable instances given a set of finding random variable instances. This algorithm is a version of the *simple bottom-up construction* algorithm given in Mahoney and Laskey (1998), adapted to the

case in which the true SSBN may be infinite. The algorithm begins with a *query set* consisting of a finite set of target random variable instances and a finite set of finding random variable instances. These are combined to construct an approximate SSBN. The approximate SSBN has an arc between a pair of random variables when one variable is a parent of the other or a context variable in its home MFrag, and there is an influencing configuration for the child variable. At each step, the algorithm obtains a new approximate SSBN by adding findings, instantiating the home MFrags of the random variables in the query set and their ancestors, adding the resulting random variable instances to the query set, removing any that are not relevant to the query, and combining the resulting set of random variable instances into a new approximate SSBN. This process continues until either there are no changes to the approximate SSBN, or a stopping criterion is met. If the algorithm is run without a stopping criterion, then if SSBN construction terminates, the resulting SSBN provides an exact response to the query or an indication that the findings are inconsistent. When the algorithm does not terminate, it defines an anytime process that yields a sequence of approximate SSBNs converging to the correct query response if one exists. In general, there may be no finite-length proof that a set of findings is consistent, but inconsistent findings can be detected in a finite number of steps of SSBN construction.

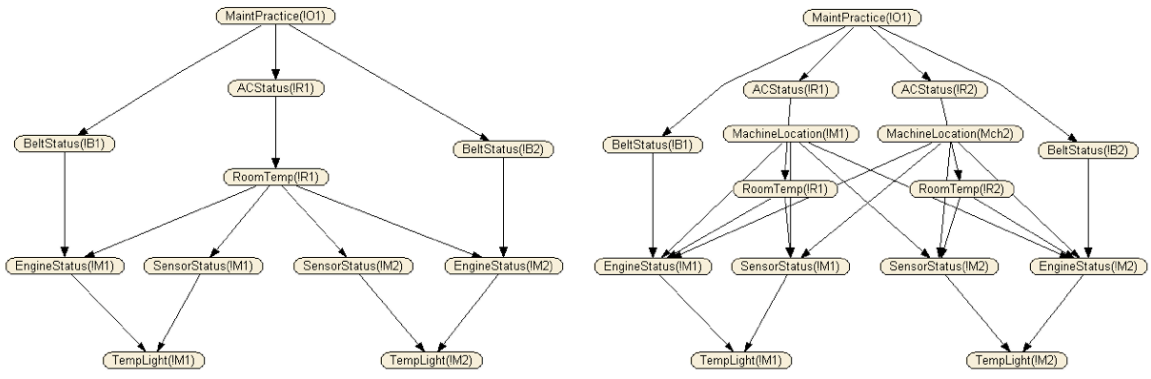
Figure 10 shows two SSBNs constructed from the MEBN theory of Figure 2 for a query on the engine status of two machines, the first for the case in which the two machines are known to be in the same room, and the second for the case in which the two machines may be in different rooms. In the first case, learning that the engine in one machine is overheated results in an increase in the probability that the other engine is overheated; in the second case, the same information has almost no effect on the probability distribution for the other machine (there is a small impact because of the influence of the evidence on beliefs about the maintenance practices of the owner).

1. *Initialization*: Set the *query set* Q_0 to the union of the target nodes and the finding nodes. Initialize the *RV instances* $\mathcal{R}_0 = Q_0$. Set the maximum number of states per random variable N_0 equal to a finite integer. Set $i = 0$.
2. *SSBN Structure Construction*. Set the current SSBN $\mathcal{B}_i$ to contain the nodes in $\mathcal{R}_i$ and all arcs corresponding to influencing configurations. Remove from $\mathcal{B}_i$ any barren nodes, nodes *d*-separated from target nodes by finding nodes, and nuisance nodes for which marginal distributions do not need to be updated.
3. *Local Distribution Construction*. Set the local distributions in $\mathcal{B}_i$, modifying the local distributions to restrict random variables to no more than N_i possible values and, to approximate the effect of random variables that have not been enumerated, and compute for no more than K_i steps.
4. *Inference*. Apply standard Bayesian network inference to compute conditional distributions for the target random variables given the finding random variables. If findings have probability zero, report that the findings are inconsistent.
5. *Instance Enumeration and Approximation Parameter Updating*. If a stopping criterion is met, output $\mathcal{B}_i$. Else add to $\mathcal{R}_i$ additional parents of random variables for which adding additional parents might change the distribution, increase N_i and K_i and return to Step 2.

Figure 9: SSBN Construction Algorithm Sketch
(*See Appendix for details*)

When there is uncertainty in the value of a context random variable, it appears explicitly in the SSBN. For example, *MachineLocation*(*m*) appears as a node in Figure 10*b*, with possible values *!R*1 and *!*R2, the rooms in which !*M*1 and *!*M2 might be located. Context random variables may be *de facto* parents of the other resident random variables in their MFrag: in this case, *EngineStatus*(*m*), *SensorStatus*(*m*), and *TempLight*(*m*). The distributions for *EngineStatus*(*m*) and *SensorStatus*(*m*) are multiplexor distributions. That is, the distribution of the child depends on the temperature of the room in which the machine is actually located and not on the temperatures of the other rooms in which it might have been located. The distribution for *EngineStatus*(*m*) depends on *MachineLocation*(*m*) only through whether or not its value is ⊥. Because it is known to have non-⊥ value in this situation, no arc is needed in the SSBN from *MachineLocation*(*m*) to *EngineStatus*(*m*). When there may be uncertainty in the context random variables, omitting the implicit arcs from context random variables to resident random variables in the MFrag drawings hides some of the complexity in a MEBN theory. Nevertheless, context random variables enable economical representations that facilitate knowledge engineering. Furthermore, inference algorithms can exploit context-specific independence to achieve computational efficiency.

As noted above, when an ordinary variable appears in a parent but not in its child, the random variable can have an unbounded number of parent instances in the constructed approximate SSBN. Each step of SSBN construction instantiates finitely many parents of any random variable. When there are infinitely many computationally relevant parent instances, additional instances are added at each step until a termination condition is reached. Even when a finite-size SSBN exists, constructing it and computing a query response is often intractable. It is typically necessary to approximate the SSBN by pruning arcs and random variables that have little influence on a query, and/or compiling parts of the SSBN to send to inference engines optimized for special problem types. The process of controlling the addition and pruning of random variable instances and arcs is called *hypothesis management*. More generally, *execution management* controls the inference process to balance accuracy against computational resources. Often, portions of an inference task can be solved exactly or approximately using efficient special-purpose reasoners. Such reasoners include constraint satisfaction systems, deductive theorem provers, differential equation solvers, heuristic search and optimization algorithms, Markov chain Monte Carlo algorithms, particle filters, etc. Online reasoning systems may interleave addition of new findings, refinement of the current approximate SSBN, computation of query responses given the current approximate



a. Two machines in the same room

b. Two machines that might or might not be in the same room

Figure 10: Situation-Specific Bayesian Networks

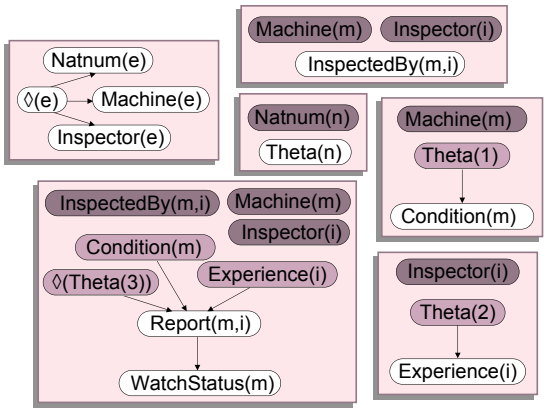
SSBN, and learning. Laskey, et al. (2000, 2001) treat hypothesis management as a problem of balancing the computational overhead of representing additional random variable instances against accuracy in responding to queries. Charniak and Goldman (1993) and Levitt et al. (1995; Binford and Levitt, 2003) also consider hypothesis management in open-world computational probabilistic reasoning systems. Hypothesis management is discussed extensively in the literature on tracking and multi-source fusion (e.g., Stone, et al., 2000).

5 Probabilistic Logics and Languages

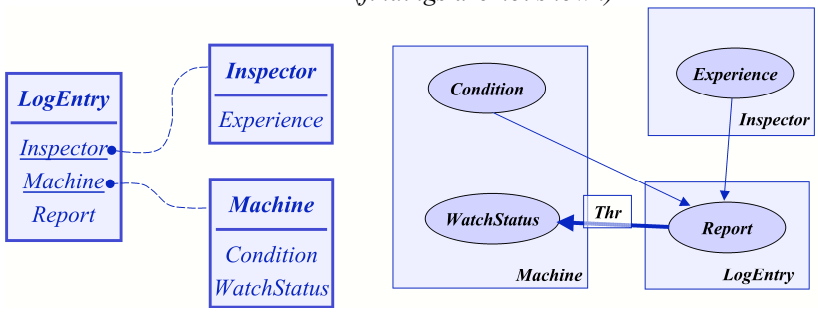
There is a growing literature on languages for representing probabilistic knowledge, the semantics of probabilistic representations, and well-foundedness, tractability and decidability of inference in probabilistic theories. The success of graphical models for parsimonious representation and tractable inference has generated strong interest in more expressive languages for reasoning with probability. Work in knowledge-based model construction (e.g., Wellman, et al., 1992) focused on constructing Bayesian networks from knowledge bases consisting of modular elements representing knowledge about small clusters of variables. Early KBMC systems were not built on decision theoretically coherent declarative domain theories, and relied on heuristic knowledge, typically encoded as procedural rules, for constructing complex models from simpler components. As work in knowledge-based model construction progressed, interest grew in the theoretical foundations of probabilistic representation languages, and in their relationship to classical first-order logic. A number of authors have investigated approaches to integrating classical logic with probability. A common approach has been to provide language constructs that allow one to express first-order theories not just about objects in a domain of discourse, but also about proportions and/or degrees of belief for statements about these objects. Bacchus et al. (1997; Bacchus, 1990) augment first-order logic with proportion expressions that represent the knowledge that a given proportion of objects in a domain have a certain property. A principle of indifference is applied to assign degrees of belief to interpretations satisfying the constraints imposed by ordinary first-order quantification and the proportion expressions. Halpern's (1991) logic can express both proportion expressions and degrees of belief, and provides a semantics relating proportions to degrees of belief. Neither of these logical systems provides a natural way to express theories in terms of modular and composable elements. Unlike Bayesian networks, which have easy to verify conditions ensuring the existence of a coherent domain theory, it is in general quite difficult in these logical systems to specify complete and consistent probabilistic domain theories, or to verify that a set of axioms is coherent.

Several languages have been developed that represent probabilistic knowledge as modular units with repeated substructures that can be composed into complex domain models. These include pattern theory (Grenander, 1996), hidden Markov models (Elliott, et al., 1995), the plates language implemented in BUGS (Gilks, et al., 1994; Buntine, 1994; Spiegelhalter, et all, 1996), object-oriented Bayesian networks (Koller and Pfeffer, 1997; Bangsø and Wuillemin, 2000; Langseth and Nielsen, 2003), and probabilistic relational models (Getoor, et al., 2000, 2001; Pfeffer, 2000). There is a great deal of commonality among languages for compactly expressing complex probabilistic domain theories (cf., Heckerman, et al., 2004). Plates in BUGS, object classes in object-oriented Bayesian networks, and PRM structures in probabilistic relational models all correspond to MFrag classes.

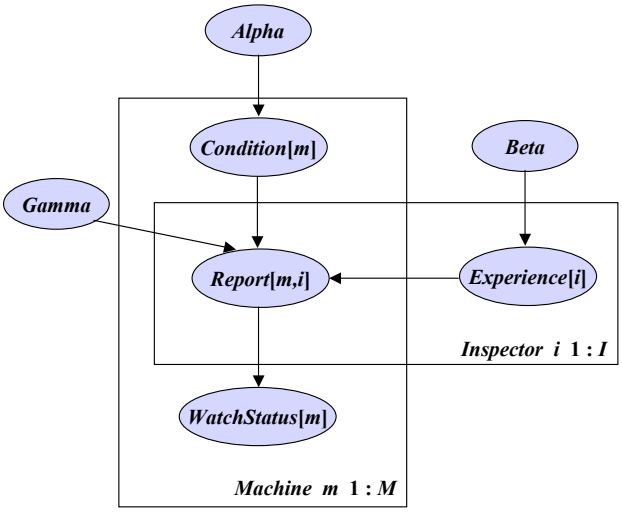
Figure 11 compares MEBN, PRM and plate representations for a theory fragment in the equipment diagnosis domain. Like Bayesian networks, plates represent a joint distribution as an acyclic directed graph in which nodes represent random variables, arcs represent direct dependence relationships, and each node is annotated with a specification of a conditional distribution of the random variable given its parents. Repeated structure in a plates model is represented by indexing repeated random variables with subscripts, and enclosing the set of random variables indexed by a given subscript in a rectangle called a "plate." These indices play



a. MEBN Fragments
(findings are not shown)



a. Probabilistic Relational Model – Relational Schema & PRM Structure
(skeleton and instances are not shown)



c. Plates

Figure 11: MFrags, PRM and Plates for Equipment Diagnosis Domain

the role of the ordinary variables in an MFrag. As in MEBN, a random variable's parents may contain indices not mentioned in the random variable, in which case the local distribution for the child random variable must specify how to aggregate influences from multiple instances of the parent random variable. Plate models are restricted to a finite number of instances of each random variable. The number of instances of each random variable is a fixed attribute of the plate model. BUGS has sophisticated capability for parameter learning, and although there is no built-in mechanism for structure learning, plate models can be constructed to represent the problem of reasoning about the presence or absence of conditional dependency relationships between random variables.

A PRM contains the following elements (Heckerman, et al., 2004; see Figure 11*b*):

- A *relational schema* that specifies the types of objects and relationships that can exist in the domain;
- A *PRM structure* that represents probabilistic dependencies and numerical probability information;
- A *skeleton* that specifies a unique identifier and a blank template for each individual entity instance;
- The *data* to fill the entries in the blank template.

Like a MEBN theory, a PRM represents a probability distribution over possible worlds. Any given PRM can be expanded into a finite Bayesian network over attributes of and relationships between the individuals explicitly represented in the skeleton. PRMs use aggregation rules to combine influences when multiple instances of a parent random variable influence a child random variable (as when multiple reports influence the *WatchStatus* random variable in Figure 11). In addition to attribute value uncertainty, PRMs have been extended to handle type uncertainty, reference uncertainty, and identity uncertainty. PRM learning theory provides a formal basis for both parameter and structure learning. Learning methods have been published (e.g., Getoor, et al., 2001) for learning both the structure and parameters of PRMs from instances in the skeleton. If the probability distribution represented by a PRM is assumed to apply to similar entities not explicitly represented in the skeleton, then PRM learning methods can be extended to allow sequential learning as new individuals are added to the skeleton over time, thus providing the logical basis for a form of open-world reasoning. One can also extend the relational schema and PRM structure "by hand" to add new entity types.

Heckerman, et al. (2004) introduce a new language, DAPER, for expressing probabilistic knowledge about structured entities and their relationships. DAPER combines the entity-relation model from database theory with directed graphical models for expressing probabilistic relationships. DAPER is capable of expressing both PRMs and plates, thus providing a unified syntax and semantics for expressing probabilistic knowledge about structured entities and their relationships. As presented in Heckerman, et al. (2004), DAPER expresses probabilistic models over finite databases, and cannot express arbitrary first-order formulas involving quantifiers. That is, DAPER is a macro language for compactly expressing finite Bayesian networks with repeated structure, and not a true first-order probabilistic logic. Because DAPER can represent PRMs and plates, this conclusion applies to these formalisms as well. On the other hand, the random variable semantics described in Section 4.1 could provide a theoretical basis for extending DAPER, and thus PRMs and plates, into a true first-order logic. Conditions could be identified under which DAPER models of unbounded cardinality express well-defined probability distributions over models. If developed more fully, the relationship sketched here between MEBN theories, PRMs and plates would facilitate construction of such an extension.

Object-oriented Bayesian networks represent entities as instances of object classes with class-specific attributes and probability distributions. Reference attributes allow representation of function composition. Although OOBNs do not have multi-place relations, these can be handled by defining new object types to represent multi-place relations. Structure and parameter learning methods for OOBNs have been developed (e.g., Bangsø, Langseth and Nielsen, 2001; Langseth and Nielsen, 2003). The current literature on OOBNs does not treat type and reference uncertainty, although clearly it would be possible to extend OOBNs to handle these kinds of uncertainty. An advantage of OOBNs is the ability to represent *encapsulated* information, or random variables defined internally to an object that are independent of external random variables given the interface random variables that shield an object from its environment. The semantics of encapsulation is based on conditional independence relationships. Thus, the concept of encapsulation could be extended to other languages based on graphical models, including MEBN theories and DAPER models with encapsulated random variables. As with plates and PRMs, the semantics described in Section 4.1 could provide a theoretical basis for extending OOBNs to achieve full first-order expressive power.

A feature of MEBN not present in PRMs, plates or OOBNs is the use of context constraints to specify logical conditions that determine whether one random variable influences another. A similar effect can be achieved by using aggregation functions that ignore influences ruled out by the context, but this is more cumbersome. PRMs and OOBNs are founded on a type system. Sophisticated implementations have subtyping, inheritance, and the ability to represent type uncertainty (e.g., IET, 2004). MEBN can be extended to a typed logic that has many of the advantages of typed relational languages (Costa 2005).

Like MEBN, relational Bayesian networks (Jaeger 1998; 2001) provide formal semantics for probability languages that extend Bayesian networks to achieve first-order expressiveness. Random variables in a relational Bayesian network are all logical. A RBN has a set of pre-defined relations used in defining the local distributions and a set of probabilistic relational symbols, which represent uncertain relations on the domain. A RBN defines a joint probability distribution on models of the uncertain relations. Probability formulas specify how to combine influences from multiple instances of the parents of a random variable to obtain a conditional distribution for the random variable given finite sets of instances of its parents. General relational Bayesian networks can represent probability distributions only over finite domains, although non-recursive RBNs have been extended to represent probability distributions over countably infinite domains (Jaeger, 1998).

Bayesian logic programs (e.g., Kersting and De Raedt, 2001; De Raedt and Kersting, 2003) also express uncertainty over interpretations of first-order theories. To ensure decidability, BLPs have typically been restricted to Horn clause theories. Bayesian logic programs and MEBN theories represent complementary approaches to specifying first-order probabilistic theories. BLPs represent fragments of Bayesian networks in first-order logic; MEBN theories represent first-order logic sentences as MFrags. Although the restriction to Horn clause logic limits the expressiveness of BLP languages, this limitation is balanced by the efficiency of algorithms specialized to Horn clause theories. Research in Bayesian logic programming is applicable to the problem of execution management in SSBN construction. That is, an execution manager can identify portions of an inference task that involve only Horn clauses, and send these to an inference engine specialized for efficient reasoning with Horn clauses. MEBN semantics could be used to develop extensions to BLP languages that could handle knowledge bases not limited to Horn clauses.

Other research on integrating logic and probability includes Poole's (2003) parameterized Bayesian networks, Ngo and Haddawy's (1997) work on context-specific probabilistic knowledge bases, PRISM (Sato, 1998), IBAL (Pfeffer, 2001), and BLOG (Milch, et al., 2005). Parameterized Bayesian networks are designed to provide the ability to reason about individuals not explicitly named, an important capability lacking in most probabilistic languages. Poole presents an algorithm for performing inference without grounding the theory. Like MEBN, random variables in a parameterized Bayesian network can take arguments; individuals in a population can be substituted for the parameters to form instances of the random variables. Like MEBN, the population over which the parameters range can be finite or infinite. Poole considers only models without recursion. Thus, a parameterized Bayesian network corresponds to a MEBN theory with no recursive links. For such theories, Poole's inference algorithm would provide an alternative, possibly more efficient, to the SSBN algorithm presented here. Ngo and Haddawy represent probabilistic knowledge as universally quantified sentences that depend on context. Like MEBN, Ngo and Haddawy exploit context constraints to focus inference on relevant portions of the knowledge base. Unlike MEBN, Ngo and Haddawy separate context, which is non-probabilistic, from uncertain hypotheses, for which context-specific probability distributions are defined. A context-sensitive knowledge base corresponds to a partially specified MEBN theory in which there is a reserved subset of logical random variables that may appear as context random variables in MFrags, but that have no home MFrags and whose truth-values are assumed to be known at problem solving time. PRISM is a logic programming language in which facts can have parameterized probability distributions. Like a MEBN theory, a PRISM program defines a probability distribution over interpretations. A PRISM program can be used as a random sampler from the distribution it defines. PRISM also supports abductive reasoning and EM learning. IBAL is a probabilistic programming language that allows users to write functional programs with stochastic branches. Given such a program, IBAL uses a variety of inference methods to provide a probability distribution over outputs of the program. Results may be conditioned on user-specified evidence. IBAL supports parameter learning and utility maximization. BLOG (Milch, et al., 2005) is a new language that enables probabilistic reasoning about unknown entities, and about domains that can contain unknown numbers of entities. Under appropriate conditions such as the ones defined in 3.3, BLOG could also express probability distributions over interpretations of a broad class of first-order theories.

Hidden Markov models, dynamic Bayesian networks and partially dynamic Bayesian networks (Bayesian networks containing both static and dynamic nodes) provide the ability to define simple recursive relationships. Pfeffer (2000) also considers recursive probabilistic models, which can express a richer class of recursive relationships. It is straightforward to express HMMs, DBNs, and recursive probabilistic models as MEBN theories (e.g., Figure 3).

Pattern theory (Grenander, 1996) is a graphical modeling language based on undirected graphs. There is an extensive literature on applications of undirected graphical models to image understanding, geospatial data, and other problems in which there is no natural direction of influence. A hybrid language could be defined that extends MEBN to permit both directed and undirected arcs. Such an extension is not considered here.

A common problem for first-order graphical probabilistic languages is how to specify local distributions when a random variable has different numbers of parents in different ground Bayesian networks corresponding to a given first-order probabilistic theory. Probabilistic relational models use aggregation functions, in which a summary statistic is computed from the instances of one of the parents, and the local distribution depends on the summary statistic. For

example, the distribution for *WatchStatus* in Figure 11 depends on a summary statistic that aggregates the total number of problematic reports received about an item. Many knowledge-based Bayesian network construction approaches use combination rules (e.g., Natarajan, 2005; Ngo and Haddawy, 1997). With combination rules, the modeler defines a probability distribution for a single instance of each of the parents of a random variable, and a combination rule that specifies how to combine these distributions when the ground model contains multiple instances of some or all of the parents. Influence counts can represent both combining rules and aggregation functions.

To show how influence counts can represent combining rules, consider an extension of our diagnosis example in which *EngineStatus*(*m*) depends on *BeltStatus*(*b*) and *GasketStatus*(*g*), and in which the context constraints specify *Isa*(*Belt*,*b*) and *Isa*(*Gasket*,*g*). To specify a combining rule, the modeler would specify a probability distribution for *EngineStatus*(*m*) given each belt/gasket configuration and each room temperature, and define a function to combine these distributions. Suppose a particular machine has two belts and three gaskets, and is located in one of two rooms. Making all legal substitutions would yield twelve probability distributions: one for each of the six belt/gasket combinations in each of the two rooms. The combining function would specify how to obtain a single distribution from these twelve distributions. To use influence counts to define a combining rule, we would simply specify a probability distribution for each parent configuration, and then use each configuration's influence count to specify the number of copies of the corresponding distribution to combine. To represent aggregation rules with influence counts is a little less straightforward. Suppose we want to define an aggregation function that depends on the total number of broken belts and the total number of broken gaskets. In our machine with two belts and three gaskets, each belt contributes to three of the six influencing configurations, and each gasket contributes to two of the six influencing combinations. Thus, we would need to divide the total influence counts for broken belt configurations by 3 and the total influence counts for broken gaskets by two, in order to obtain the needed aggregation function. This simple rule breaks down when there are context constraints that rule out some of the combinations. For example, one of the belts might be allowed to combine with only one of the gaskets. This case could be handled by making the entity identifier $\Diamond(e)$ a parent and counting the number of unique belt instances and unique gasket instances among the influencing configurations.

One concern that may arise with representing combining rules and aggregates is whether Conditions 3*c* and 3*e* are satisfied. For example, the average of infinitely many terms does not in general satisfy 3*c*. In fact, a condition like 3*c* is required precisely because in its absence, an average of infinitely many terms may not have a well-defined distribution. Most other formalism avoid this difficulty by assuming the domain is finite. In BLOG, the cardinality may be unbounded, but it is assumed to be finite. In finite domains, Conditions 3*c* and 3*e* are always satisfied. This is because in each interpretation, all but finitely many entity identifiers have value $\perp$. Therefore, there are finitely many influencing configurations for any random variable. The number of influencing configurations may vary from interpretation to interpretation, but this is allowable under Condition 3*c*. Thus, all random variables, even aggregates and combined influences, have well-defined distributions, because Conditions 3*c* and 3*e* are satisfied. Indeed, these conditions are satisfied in infinite domains under the assumption that in any interpretation, each random variable is actually influenced by only a finite (possibly unbounded) number of its parents. This level of expressiveness is more than sufficient to represent any problem a knowledge engineer is likely to encounter. These conditions impose real restrictions only with the

attempt to represent completed infinities (e.g., the average of an infinite number of non-zero numbers). While appropriate relaxations of these conditions are of theoretical interest, identifying conditions to ensure that MEBN theories can represent completed infinities is not a pressing practical issue.

6 Summary and Discussion

Graphical models were initially limited to problems in which the relevant random variables and relationships could be specified in advance. Languages based on graphical models are rapidly reaching the expressive power required for general computing applications. It is becoming possible to base computational inference and learning systems on rationally coherent domain models implicitly encoded as sets of graphical model fragments, and to use such coherent deep structure models to guide reasoning and knowledge discovery. Probability theory provides a logically coherent calculus for combining prior knowledge with data to evolve an agent's knowledge as observations accrue. Probability theory also provides a principled approach to knowledge interchange among different reasoners.

This paper presents a first-order Bayesian language called Multi-Entity Bayesian Networks (MEBN). The syntactic similarity of MEBN to standard first-order logic notation clarifies the relationship between first-order logic and probabilistic logic. A MEBN theory (MEBN theory) assigns probabilities to models of an associated FOL theory. MEBN theories partition FOL theories into equivalence classes of theories with the same logical content but different probabilities assigned to models. Provable statements in FOL correspond to statements in the associated MEBN theory for which SSBN construction terminates with a probability of 1 assigned to the value T. A MEBN theory corresponding to an inconsistent FOL theory has at least one finding equal to $\perp$ with probability 1. If the associated MEBN theory is inconsistent, SSBN can determine in finitely many steps that it is inconsistent. When SSBN construction does not terminate but the MEBN theory represents a globally consistent joint distribution, the construction process gives rise to an anytime sequence of approximations that converges in the infinite limit to the correct response to the query.

MEBN can represent a very large class of probability distributions, and can be used to construct domain theories for a wide variety of application domains. The conditions in Definitions 3 and 5 do impose some constraints on the distributions MEBN can represent. For example, MEBN cannot represent the average of infinitely many non-zero numerical random variables. It cannot represent random variables with infinitely long ancestor chains, such as a Markov chain that extends without bound in both the past and the future directions. Nevertheless, MEBN provides a rich language for knowledge representation. It can handle n-ary relations, context-specific independence, quantifiers, combining functions, and aggregates.

The unit of representation in MEBN is a conceptually meaningful cluster of related random variables. In many applications, this representational unit is more natural than a focus on uncertainty about attributes of objects. Unlike OOBNs and PRMs, MEBN can represent *n*-ary relationships. Although the specification given in this paper is untyped, the examples illustrate a simple type system that was formalized in Costa (2005). It is straightforward in typed MEBN to represent probabilistic relational models. Because there presently is no direct MEBN implementation, several published applications have translated MEBN theories into relational models and used the Quiddity*Suite probabilistic relational modeling and KBMC toolkit (IET, 2004) to construct situation-specific Bayesian networks (e.g., Costa, et al., 2005; AlGhamdi, et al., 2005). Rules are given in Costa (2005) for translating MEBN theories into Quiddity*Suite

frames. There are some features of MEBN (most notably context constraints) that cannot be represented declaratively in standard relational languages, but the ability of Quiddity*Suite to combine Prolog-style rules with a frame-based relational modeling language provides the ability to specify much more powerful declarative representations (e.g., Fung, et al., 2005).

Many languages designed for tractable implementation have taken the strategy of restricting expressiveness to ensure that answers to probabilistic queries are decidable. In an open world, the answer to many queries of interest will be undecidable, and the best that can be expected is an approximate answer. Languages that provide decidable, closed-form responses to limited classes of queries have an important place both theoretically and practically. Nevertheless, intelligent reasoning in a complex world requires principled methods of coping with intractable and even undecidable problems. MEBN exploits the language of graphical models to compose consistent domain theories out of modular components connected via clearly defined interfaces, and thus can support efficient implementations of tractable domain theories. Yet, MEBN can represent highly complex, intractable, and even undecidable domain theories. Although the answer to a probabilistic query may be undecidable, and may be intractable even when it is decidable, Bayesian decision theory provides a sound mathematical basis for defining and analyzing the properties of processes that converge to the correct response to undecidable queries, as well as resource-bounded processes that balance efficiency against accuracy. Bayesian theory also provides semantics for the relationship between empirical proportions and probabilities, and logically justified and theoretically principled way to combine empirical frequencies with prior knowledge to refine theories in the light of observed evidence.

MEBN is inherently open. Bayesian learning theory provides an inbuilt capability for MEBN-based systems to learn better representations as observations accrue. Parameter learning can be expressed as inference in MEBN theories that contain parameter random variables. Structure learning can also be handled by introducing multiple versions of random variables having home MFrags with different structures. A more natural approach to structure learning, as well as a more flexible type system, requires a polymorphic extension of MEBN. Clearly, a typed MEBN with polymorphism would be desirable for many applications. We chose in this paper to focus on the basic version of the logic to highlight its relationship to classical first-order logic and demonstrate that the logic is sufficiently powerful to represent general first-order theories. Extensions of MEBN are planned to incorporate additional expressivity.

Appendix A: Proofs and Algorithms

This appendix proves that a MEBN theory represents a globally consistent joint distribution over random variable instances, proves that a MEBN theory constructed as described in Section 4.2 places non-zero probability of value T on logical random variables corresponding to satisfiable first-order sentences, presents the SSBN construction algorithm, shows that SSBN construction identifies an unsatisfiable set of findings in finitely many steps, and proves that when findings are consistent, SSBN construction converges with probability 1 to the posterior distribution over a MEBN theory's random variables given that all finding random variables have value T.

A.1. Proof of Existence Theorem

***Theorem 1*:** Let $\mathcal{T}' = \{ \mathcal{F}_1, \mathcal{F}_2 \ldots \}$ be a simple MEBN theory. There exists a joint unique probability distribution $\mathcal{P}^{\text{gen}}_{\mathcal{T}'}$ on the set of instances of the random variables of its MFrags that is consistent with the local distributions assigned by the MFrags of $\mathcal{T}'$. This distribution respects the independence assumptions encoded in the MFrags. That is, a random variable instance is

conditionally independent of its non-descendants given its full (possibly infinite) set of parent instances.

***Proof*:** Let $\mathcal{Z}=\{\phi_1(\alpha_1), \ldots, \phi_m(\alpha_m)\}$ be a finite subset of $\mathcal{N}_{\mathcal{T}}$, and let D = max $[d_{\phi(\alpha)} : \phi(\alpha)\in \{\phi_1(\alpha_1), \ldots, \phi_m(\alpha_m)\}\ ]$ be the maximum depth of the instances of $\mathcal{Z}$. Suppose $D = 0$. Let $\pi_{\mathcal{T}1}(\phi_1(\alpha_1), \ldots, \phi_m(\alpha_m))$ be a distribution in which the $\phi_i(\alpha_i)$ are independent and distributed according to the default distributions $\pi_{\phi_i(\theta_i)}(\bullet \mid \varnothing)$ from their home MFrags $\mathcal{F}_{\phi_i(\alpha_i)}$. All finite-dimensional distributions constructed in this way from depth 0 elements of $\mathcal{N}_{\mathcal{T}}$ are consistent with each other and with the local distributions of $\mathcal{T}$. Therefore, Kolmogorov's existence theorem[11] implies that these finite-dimensional distributions can be extended to a joint distribution $\pi_{\mathcal{T}1}$ over all instances of depth zero random variables, and this joint distribution is consistent with the local distributions of $\mathcal{T}$.

Now, suppose $\mathcal{T}$ represents a joint distribution $\pi_{\mathcal{T}D}$ over all instances of all random variables of depth less than D. Let $\mathcal{Z} = \{\phi_1(\alpha_1), \ldots, \phi_m(\alpha_m)\}$ be a finite subset of $\mathcal{N}_{\mathcal{T}}$ such that no $\phi_i(\alpha_i) \in \mathcal{Z}$ has depth greater than D. Let $\mathcal{A}$ denote the (possibly infinite) subset of $\mathcal{N}_{\mathcal{T}}$ consisting of the ancestors of depth D elements of $\mathcal{Z}$, together with any elements of $\mathcal{Z}$ with depth strictly less than D. Clearly, any instance $\varphi(\beta)\in\mathcal{A}$ must have depth less than D. Therefore, the marginal distribution of $\pi_{\mathcal{T}D}$ represents a joint distribution for $\mathcal{A}$ consistent with the local distributions of $\mathcal{T}$.

Let $S=\{\varphi(\beta)=\gamma : \varphi(\beta)\in\mathcal{A}\}$ be a set of value assignment terms, one for each element of $\mathcal{A}$. Suppose $\phi_i(\alpha_i) \in \mathcal{Z}$. If $\phi_i(\alpha_i)$ has depth less than D, then $\phi_i(\alpha_i)\in\mathcal{A}$ and S assigns a particular value to $\phi_i(\alpha_i)$ with probability 1. Otherwise, condition 3*e* implies that there is a finite subset $S_{\phi_i(\alpha_i)}\subset S$ such that $\pi_{\phi_i(\alpha_i)}(\bullet \mid S_{\phi_i(\alpha_i)})=\pi_{\phi_i(\alpha_i)}(\bullet \mid S^*)$ whenever $S_{\phi_i(\alpha_i)}\subset S^*\subset S$.[12] Thus, given the value assignments in S, $\mathcal{T}$ assigns a well-defined conditional distribution to each $\phi_i(\alpha_i) \in \mathcal{Z}$, which is denoted $\pi_{\phi_i(\alpha_i)}(\bullet \mid S)$. Define a joint conditional distribution

$$\pi_{\mathcal{T}(D+1)}(\phi_1(\alpha_1)=\gamma_1, \ldots, \phi_m(\alpha_m)=\gamma_m \mid S) = \prod_{i=1}^{m} \pi_{\phi_i(\alpha_i)}(\phi_i(\alpha_i)=\gamma_i \mid S).$$

in which the $\phi_i(\alpha_i)$ are independent and distributed as assigned by the local distributions in their home MFrags conditional on the value assignments in S. Existence of both a joint conditional distribution for the $\phi_i(\alpha_i)$ and a marginal distribution for S implies that the marginal joint distribution

$$\pi_{\mathcal{T}(D+1)}(\phi_1(\alpha_1), \ldots, \phi_m(\alpha_m)) = \int \prod_{i=1}^{m} \pi_{\phi_i(\alpha_i)}(\phi_i(\alpha_i) \mid S) d\pi_{\mathcal{T}D}(S). \qquad (1)$$

exists and is consistent with the local distributions of $\mathcal{T}$. The marginal distribution (1) is expressed as an integral rather than a sum because there may be uncountably many different ways to choose the value assignments $S=\{\varphi(\beta)=\gamma : \varphi(\beta)\in\mathcal{A}\}$.

This construction can be carried out for any finite set of depth D instances, and it is clear that all the distributions thus defined are consistent with each other and with the local distributions of $\mathcal{T}$. This implies that $\mathcal{T}$ represents a joint distribution over arbitrary finite subsets of $\mathcal{N}_{\mathcal{T}}$, and that the distributions constructed in this way are consistent with each other and with the local

[11] Kolmogorov's existence theorem (c.f., Billingsley, 1995) states that if joint distributions exist for all finite subsets of a collection of random variables, and if all these finite-dimensional distributions are consistent with each other, then a joint distribution exists for the infinite collection of random variables.

[12] Theorem 1 holds under weaker conditions on the local distributions, but condition 3*e* suffices to show that MEBN can represent classical first-order logic.

distributions of $\mathcal{T}$. A second application of Kolmogorov's existence theorem implies that $\mathcal{T}$ represents a joint distribution over all instances of random variables in $\mathcal{V}_{\mathcal{T}}$. It is clear that this distribution is consistent with the local distributions of $\mathcal{T}$. Uniqueness and satisfaction of the causal Markov conditions are immediate consequences of the construction of the distribution. ■

A.2. SSBN Construction Algorithm

The situation-specific Bayesian network construction algorithm takes a MEBN theory $\mathcal{T}$, a finite (possibly empty) set of *target* random variable instances, and a finite (possibly empty) set of *finding* random variable instances, and computes a sequence of Bayesian networks containing the target and finding random variable instances. The algorithm may be interrupted at any time to obtain an approximate SSBN. If the findings are inconsistent and the algorithm is not interrupted, it will discover the inconsistency in finitely many steps. If the algorithm terminates without interruption and the findings are consistent, the last Bayesian network in the sequence can be used to compute the joint distribution of the target random variable instances given that all finding random variable instances have value T. That is, additional model construction would not change the result of the query. For some problems, the algorithm will not terminate unless it is interrupted, but it produces a sequence of approximate SSBNs that converge to the correct query response.

We give the SSBN construction for simple MEBN theories only. The modification for mixture MEBN theories is straightforward. SSBN construction proceeds as follows:

SSBNConstruct: The inputs to SSBNConstruct are:

- A simple MEBN theory $\mathcal{T}$ with partial ordering $\preceq$ and modeler-defined MFrags $\mathcal{F}$ defined on a set $\mathcal{X}$ of random variable symbols and a set $\mathcal{A}$ of constant symbols;
- A finite (possibly empty) set $\{\tau_i\}_{i\leq T}$ of non-finding random variable instances called the *target* random variable instances;
- A finite (possibly empty) set $\{\phi_i\}_{i\leq F}$ of *finding* random variable instances.

The steps in SSBNConstruct are:

1. *Initialization*. Set $Q = \{\tau_i\}_{i\leq T} \cup \{\phi_i\}_{i\leq F}$, and set $\mathcal{R}_0 = Q$. Let N_0 and K_0 be positive integers. Set the iteration number i equal to 0.
2. *SSBN structure construction*. Set the structure of the approximate SSBN $\mathcal{B}_i$ as follows (see Figure 12):
   - Set $\mathcal{B}_i$ equal to a Bayesian network in which the nodes are the random variables in $\mathcal{R}_i$. Add an arc from random variable α to β if α is an instance of a parent of β or is a context random variable in the home MFrag of β. Remove any arcs to β if there are no influencing configurations for β (i.e., there are no configurations of its parents and context random variables that match the context constraints).
   - Do until no more changes to $\mathcal{B}_i$ occur (see Figure 12):
     - Remove from $\mathcal{B}_i$ all *barren* nodes, that is, nodes having no descendants in Q;
     - Remove from $\mathcal{B}_i$ all nodes that are *d-separated* by finding nodes from any target nodes;
     - Remove from $\mathcal{B}_i$ the parents of any *nuisance node* for which there is a current cached marginal distribution. A nuisance node (Lin and Druzdzel, 1997) is a node that is

computationally relevant given the query, but is on no evidential trail[13] between an evidence and a target node.

3. *Local distribution construction*. Calculate the local distributions in $\mathcal{B}_i$ from the local distributions in the MFrags of $\mathcal{T}$, with modifications to restrict random variables to have no more than N_i possible values, to approximate the effects of random variables that have not been enumerated, and to ensure that computation of local distributions halts. Specifically:
   - If ψ is a nuisance node with a current cached marginal distribution (in this case, Step 2 above ensures that ψ will be a root node in $\mathcal{B}_i$), assign it the cached marginal distribution.
   - For any other node ψ in $\mathcal{B}_i$, let S_ψ be a configuration of states of the parents of ψ in $\mathcal{B}_i$ (by convention, $S_\psi=\varnothing$ if ψ has no parents in $\mathcal{B}_i$).
     - If S_ψ assigns each parent ψ in $\mathcal{B}_i$ to its 1st, 2nd, …, or N_i-1st state, then run the algorithm for computing the probabilities of the first N_i possible values for ψ given S_ψ, terminating the computation after K_i steps. Assign the first $N_i - 1$ states of ψ to the probabilities returned by this algorithm, and assign the N_i^{th} possible value equal to 1 minus the sum of the probabilities for the other values.
     - If S_ψ assigns any parent ψ in $\mathcal{B}_i$ to its N_i^{th} state, then assign ψ a default distribution that gives non-zero probability to all states of ψ (i.e., to all states if there are fewer than N_i or to the first N_i states otherwise).
4. *Inference*. Apply a standard Bayesian network inference algorithm to compute the conditional distribution for the non-finding random variables in $\mathcal{B}_i$ given the finding random variables in $\mathcal{B}_i$. For each node β in $\mathcal{B}_i$, cache its marginal distribution and mark it current.
   - If the inference algorithm indicates that the findings are inconsistent, then set the SSBN S equal to $\mathcal{B}_i$, output S, and stop with an indication that SSBN construction terminated and $\mathcal{T}$ is inconsistent.
   - Else, if all computationally relevant random variables have been added, no random variable in $\mathcal{B}_i$ has more than N_i possible values, and no local distribution computation terminated prior to completion, then set the SSBN S equal to $\mathcal{B}_i$; return $\mathcal{B}_i$ and the joint distribution of the target random variables; and stop with a flag indicating that SSBN construction terminated and $\mathcal{T}$ is consistent.
   - Else, go to Step 5.
5. *Instance enumeration and approximation parameter updating*. This step enumerates additional instances of random variables and increases the limits on the number of allowable states per random variable and computational steps for local distributions.

[13] A node is computationally relevant if it remains after iteratively removing all barren and *d*-separated nodes. An evidential trail between two sets of nodes is a minimal active undirected path from a node in one set to a node in the other. If a global joint distribution exists, then nuisance nodes can be marginalized out without affecting the result of the query.

- If the stopping criterion is met, output $\mathcal{B}_i$ and the joint distribution computed in Step 4, and stop with an indication that SSBN construction did not terminate.
- Else, set $\mathcal{R}_{i+1} = \mathcal{R}_i$. For each random variable instance $\beta \in \mathcal{B}_i$ for which a change in the local distribution may occur if additional parents are added, add a finite number of instances of parents of β to $\mathcal{R}_{i+1}$, using a process that ensures eventual addition of all instances of parents of β. (Here, a context random variable in a random variable's home MFrag counts as a parent.)
- Set N_{i+1} and K_{i+1} to positive integers strictly greater than N_i and K_i, respectively.
- For any node in which (*i*) new parents have been added, or (*ii*) new states of an ancestor have been added, or (*iii*) the computation did not halt in computing the local distribution of the node or one of its ancestors, mark its marginal distribution as not current.
- Increment i, and go to Step 2.

It is well known that if a set of sentences in FOL is unsatisfiable, then there exists a finite set of ground instances of a set of logically equivalent sentences that is also unsatisfiable (see, for example, Russell and Norvig, 2002; Enderton, 2001). The SSBN construction algorithm produces a sequence of Bayesian networks, each of which can be translated into a set of constraints on truth-values of a finite set of ground instances of FOL sentences implied by the MEBN theory $\mathcal{T}$. Each of these Bayesian networks encodes a probability distribution that assigns non-zero probability to any assignment of truth-values consistent with the constraints it encodes. Each approximate SSBN includes all constraints represented in the preceding approximate SSBNs, together with additional constraints. If the query set contains only the findings, then eventually all logical constraints implied by the findings and their predecessors in the random variable instance partial order are enumerated. If the set of all logical constraints is unsatisfiable, then so is a finite subset, and eventually the constraints encoded in the SSBN will include a finite unsatisfiable subset.

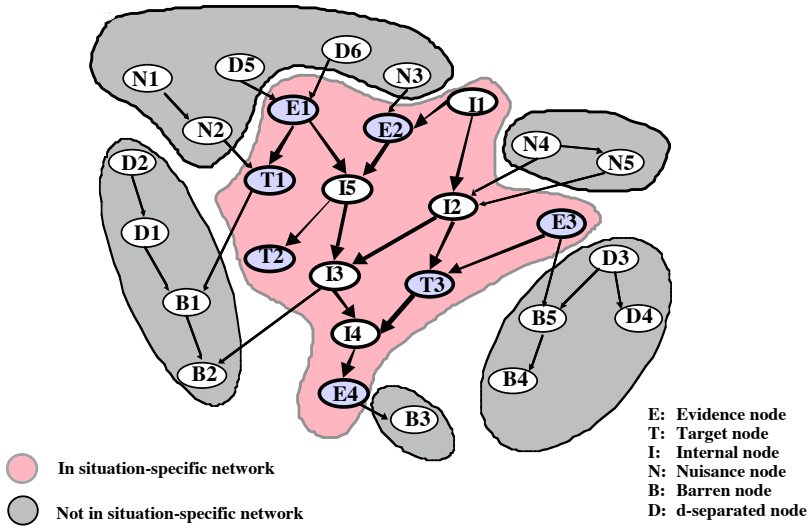


Figure 12: Situation-Specific Bayesian Network

The following theorem states that an inconsistent theory can be discovered in a finite number of steps of SSBN construction by specifying a query set consisting of only the findings, and setting SSBN construction never to stop unless an inconsistency is found.

***Theorem 6*:** If the logical constraints represented by $\mathcal{T}$ are unsatisfiable and Step 5 of *SSBNConstruct* is set never to stop, then SSBN construction on a query set consisting only of the findings of $\mathcal{T}$ terminates in finitely many steps with an indication that $\mathcal{T}$ is inconsistent.

***Proof*:** Each approximate SSBN $\mathcal{B}_i$ represents a probability distribution over interpretations of a theory for which the logical axioms form a subset of the logical axioms of $\mathcal{T}$. The domain of this interpretation is a finite set consisting of all possible assignments of values to the random variables of $\mathcal{B}_i$ such that all finding random variables have value T. The approximate SSBN $\mathcal{B}_i$ assigns non-zero probability to the hypothesis that all finding random variables have value T if and only if there is at least one interpretation on this finite domain that satisfies all the logical axioms represented in $\mathcal{B}_i$, which in turn is the case if and only if the logical axioms represented in $\mathcal{B}_i$ are simultaneously satisfiable. For $k>i$, the approximate SSBN $\mathcal{B}_k$ includes all logical constraints included in $\mathcal{B}_i$, along with any additional constraints implied by the local distributions of random variables appearing in $\mathcal{B}_{i+1}$ but not in $\mathcal{B}_i$. The SSBN construction process eventually adds all computationally relevant random variables, and therefore eventually includes all logical constraints represented by the local distributions of any random variable instances that are either findings or ancestors of findings in the random variable partial ordering $\preceq$. Thus, if the findings are unsatisfiable, eventually there will be an approximate SSBN in the sequence that represents an unsatisfiable set of constraints. ■

Note that SSBN construction will never add random variables *d*-separated from the target random variables by findings. Therefore, if the query set contains non-finding target random variables, then inconsistencies that would be introduced only by adding *d*-separated random variables will not be discovered. It is often asserted in logic texts that an inconsistent theory is "useless" because anything can be proven from a contradiction. In practice, though, inconsistent theories can be quite useful. MEBN can be used to reason with inconsistent theories, as long as queries are structured so that the target of any given query is *d*-separated by a subset of the findings from any findings that contradict this subset. Thus, MEBN may turn out to be a useful tool for studying conditions under which inconsistent theories can provide accurate results to probabilistic queries.

Condition 3*e* of Definition 3 implies that in any possible world, each local distribution can be computed from finitely many instances of the random variable's parents and context random variables. However, which instances are needed can vary from possible world to possible world, and there may be no upper bound on how many instances are needed. To show that SSBN construction converges to the correct result, it is necessary to show that the correct response to a query can be approximated to arbitrary accuracy by explicitly representing only a finite number of random variable instances.

***Lemma 7*:** Let $Q = \{\theta_i\}_{i \leq M}$ be a finite set of random variable instances from a MEBN theory $\mathcal{T}$. Let $\mathcal{R}$ denote the set of all random variable instances that are elements of Q or ancestors of elements of Q. Let $\mathcal{R}_0 \subset \mathcal{R}_1 \subset \mathcal{R}_2 \subset \cdots$ be an increasing sequence of finite sets of random variable instances such that $Q = \mathcal{R}_0$ and $\mathcal{R} = \bigcup_i \mathcal{R}_i$. Let $\mathcal{B}_i$ be the Bayesian network constructed from the random variables in $\mathcal{R}_i$. That is: (*i*) the nodes of $\mathcal{B}_i$ are the random variable instances in $\mathcal{R}_i$; (*ii*)

there is an arc from θ_i to θ_j if θ_i is either a parent of θ_j or a context random variable in its home MFrag, and if there is at least one influencing configuration containing a value assignment for θ_j; and (*iii*) the local distribution for each θ_i is given by its local distribution π_{θ_i} in its home MFrag. Let $S = \{\theta_i=\alpha_i\}_{i\leq M}$ be an assignment of values to the random variables in Q. Then $P_{\mathcal{B}_i}(S)$ converges to $P^{\text{gen}}_{\mathcal{T}}(S)$ as $i\to\infty$.

***Proof*:** The proof is by induction on the maximum depth of random variable instances in Q. Clearly, the result holds if all instances in Q are of depth zero. Suppose the result holds for all random variable instances of depth less than D. Suppose the maximum depth of random variables in Q is D.

Let $\mathcal{R}_i^0$ be the set obtained by removing from $\mathcal{R}_i$ all random variables of depth D; and let Q^0 = $\mathcal{R}_0^0$. Let S^0 be an assignment of values to random variables in Q^0 that agrees with S on the random variables in $Q\cap Q^0$ (that is, the random variables in Q of depth strictly less than D). Let $\mathcal{B}_i^0$ be the Bayesian network constructed as described above from the random variables in $\mathcal{R}_i^0$. By the induction hypothesis, $P_{\mathcal{B}_i^0}(S^0)\to P^{\text{gen}}_{\mathcal{T}}(S^0)$ as $i\to\infty$.

Let $\mathcal{U}_i = \mathcal{R}_i\backslash Q$. That is, $\mathcal{U}_i$ consists of random variables in $\mathcal{B}_i$ that are not in Q, and let $\mathcal{U}_\infty$ = $\bigcup_i \mathcal{U}_i$. Let $\mathcal{X}_\infty$ denote an assignment of values to the random variable instances in $\mathcal{U}_\infty$, and let $\mathcal{X}_i$ denote the subset of value assignments corresponding to random variables in $\mathcal{U}_i$. Suppose none of the depth D random variables in $\mathcal{R}_i$ has value $\perp$. Let θ be a depth D random variable. Condition 3*e* of Definition 3 implies that there is an integer N such that:

$$\pi_\theta(\alpha|\mathcal{X}_N\cup S^0) = \pi_\theta(\alpha|\mathcal{X}_{N+1}\cup S^0) = \cdots = \pi_\theta(\alpha|\mathcal{X}_\infty\cup S^0)\,. \tag{2}$$

Let N^* denote the smallest N for which (2) holds. The number N^* is a function of $\mathcal{X}_\infty\cup S^0$. Marginalized over $\mathcal{X}_\infty$, N^* has a probability distribution $P^{\text{gen}}_{\mathcal{T}}(N^*\mid S^0)$.

We can write:

$$\begin{aligned} &P^{\text{gen}}_{\mathcal{T}}(S) \\ &= \sum_n \sum_{\mathcal{X}_n} \left(\prod_{\substack{(\theta=\alpha)\in S \\ \text{depth}(\theta)=D}} \pi_\theta(\theta=\alpha\mid \mathcal{X}_n\cup S^0) P^{\text{gen}}_{\mathcal{T}}(\mathcal{X}_n\mid S^0, N^*=n) \right) P^{\text{gen}}_{\mathcal{T}}(N^*=n\mid S^0) P^{\text{gen}}_{\mathcal{T}}(S^0) \end{aligned} \tag{3}$$

Let $P^{n^*}_{\mathcal{T}}(S)$ be an approximation of $P^{\text{gen}}_{\mathcal{T}}(S)$ obtained by enumerating only the finite set $\mathcal{U}_{n^*}\cup Q$ of random variables:

$$P^{n^*}_{\mathcal{T}}(S) = \sum_{\mathcal{X}_{n^*}} \prod_{\substack{(\theta=\alpha)\in S \\ \text{depth}(\theta)=D}} \pi_\theta(\theta=\alpha\mid \mathcal{X}_{n^*}\cup S^0) P^{\text{gen}}_{\mathcal{T}}(\mathcal{X}_{n^*}\cup S^0) \tag{4}$$

Combining (3) and (4), and noting that $\pi_\theta(\theta=\alpha\mid\mathcal{X}_{N^*}\cup S^0) = \pi_\theta(\theta=\alpha\mid\mathcal{X}_{n^*}\cup S^0)$ when $N^*\leq n^*$, we have:

$$\begin{aligned} &\left|P^{n^*}_{\mathcal{T}}(S) - P^{\text{gen}}_{\mathcal{T}}(S)\right| \\ &\quad= \sum_{n>n^*}\sum_{\mathcal{X}_n} \left(\prod_{\substack{(\theta=\alpha)\in S \\ \text{depth}(\theta)=D}} \pi_\theta(\theta=\alpha\mid \mathcal{X}_n\cup S^0) P^{\text{gen}}_{\mathcal{T}}(\mathcal{X}_n\cup S^0\mid N^*=n) \right) P^{\text{gen}}_{\mathcal{T}}(N^*=n, S^0) \\ &\quad\leq P^{\text{gen}}_{\mathcal{T}}(N^*\geq n^*\mid S^0) P^{\text{gen}}_{\mathcal{T}}(S^0)\,. \end{aligned} \tag{5}$$

Let u be a positive real number, and let n^* be an integer such that $\sum_{n>n^*} \mathcal{P}_{\mathcal{T}}^{\text{gen}}(N^* = n) < u/2$. Then $\left|\mathcal{P}_{\mathcal{T}}^{n^*}(S) - \mathcal{P}_{\mathcal{T}}^{\text{gen}}(S)\right| < (u/2)\mathcal{P}_{\mathcal{T}}^{\text{gen}}(S^0)$. By the induction hypothesis, the distributions $\mathcal{P}_{\mathcal{B}_i^0}(\mathcal{X}_{n^*} \cup S^0)$ converge to $\mathcal{P}_{\mathcal{T}}^{\text{gen}}(\mathcal{X}_{n^*} \cup S^0)$ as $i \to \infty$. We can therefore choose n^* sufficiently large that:

$$\left|\mathcal{P}_{\mathcal{B}_i}(S) - \mathcal{P}_{\mathcal{T}}^{n^*}(S)\right| = \left|\sum_{\mathcal{X}_{n^*}} \prod_{\substack{(\theta=\alpha)\in S \\ \text{depth}(\theta)=D}} \pi_\theta(\theta = \alpha \mid \mathcal{X}_{n^*} \cup S^0)\left(\mathcal{P}_{\mathcal{B}_i^0}(\mathcal{X}_{n^*} \cup S^0) - \mathcal{P}_{\mathcal{T}}^{n^*}(\mathcal{X}_{n^*} \cup S^0)\right)\right|$$
$$< (u/2)\mathcal{P}_{\mathcal{T}}^{\text{gen}}(S^0).$$

Then for $i > n^*$:

$$\left|\mathcal{P}_{\mathcal{B}_i}(S) - \mathcal{P}_{\mathcal{T}}^{\text{gen}}(S)\right| \leq \left|\mathcal{P}_{\mathcal{B}_i}(S) - \mathcal{P}_{\mathcal{T}}^{n^*}(S)\right| + \left|\mathcal{P}_{\mathcal{T}}^{n^*}(S) - \mathcal{P}_{\mathcal{T}}^{\text{gen}}(S)\right| < u\mathcal{P}_{\mathcal{T}}^{\text{gen}}(S^0). \quad (6)$$

Therefore, $\mathcal{P}_{\mathcal{B}_i}(S)$ converges to $\mathcal{P}_{\mathcal{T}}^{\text{gen}}(S)$.

Now consider the case in which one or more of the depth D random variables has value $\perp$. It is clear that if $\mathcal{P}_{\mathcal{B}_i}(S)$ converges to $\mathcal{P}_{\mathcal{T}}^{\text{gen}}(S)$ for all S in which k or fewer of the depth D random variables has value $\perp$, then it must also converge when k+1 of the depth D random variables has value $\perp$. This establishes the result for sets Q of depth no greater than D, and thus concludes the proof. ■

***Theorem 8*:** Suppose the logical constraints represented by $\mathcal{T}$ are satisfiable. Furthermore, suppose that the algorithm described in Definition 3*c* for computing values of $\pi_{\psi(\varepsilon)}(A|S)$ returns a zero value only if the exact value $\pi_{\psi(\varepsilon)}(A|S)$ is equal to zero. If Step 7 of *SSBNConstruct* is set never to stop, then SSBN construction on query set Q either terminates with the distribution $\mathcal{P}_{\mathcal{T}}^{\text{gen}}(Q_0 \mid \{\phi_i\}_{i=i\leq F})$, or produces a sequence $\mathcal{B}_1, \mathcal{B}_2, \ldots$, in which the probability distribution for Q_0 given the findings in $\mathcal{B}_i$ converges to the distribution represented by $\mathcal{T}$.

***Proof*:** Lemma 7 establishes that the distribution on Q can be approximated to arbitrary accuracy by enumerating only finitely many of the random variable instances enumerated during SSBN construction. However, unlike in Lemma 7, SSBN construction also approximates the local distributions by enumerating only finitely many possible values and terminating computation after a finite of steps. Because the maximum number of possible values and the maximum length of computation increase with the number of SSBN steps, and do not have an upper bound, these additional approximations can be added without affecting convergence. ■

Acknowledgements

Research for this paper was partially supported by DARPA & AFRL contract F33615-98-C-1314, Alphatech subcontract 98036-7488. Additional support was provided by the Advanced Research and Development Activity (ARDA), under contract NBCHC030059, issued by the Department of the Interior. The views, opinions, and findings contained in this paper are those of the author and should not be construed as an official position, policy, or decision, of DARPA or ARDA unless so designated by other official documentation. Appreciation is extended to Bruce D'Ambrosio, Suzanne Mahoney, Mike Pool, Bikash Sabata, Masami Takikawa, Dan Upper, and Ed Wright for many helpful discussions. Special thanks are due to Paulo Costa and Tod Levitt for extensive feedback on earlier drafts. The author is grateful to the anonymous reviewers of earlier drafts for their thorough reviews, insightful comments, and useful suggestions. Special thanks are due to an anonymous reviewer of a previous version of this paper for finding a few errors.

References

Alghamdi, G., Laskey, K.B., Wright, E., Barbara, D., and Chang, K.-C., 2005. "Modeling Insider Behavior Using Multi-Entity Bayesian Networks." *10th Annual Command and Control Research and Technology Symposium*.

Bacchus, F., 1990. "Representing and Reasoning with Probabilistic Knowledge: A Logical Approach to Probabilities." Boston, MA, MIT Press.

Bacchus, F., Grove, A., Halpern, J.Y., and Koller, D., 1997. "From statistical knowledge bases to degrees of belief." *Artificial Intelligence*, Vol. 87: 75-143

Bangsø, O., Langseth, H., and Nielsen, T., 2001. "Structural Learning in Object Oriented Domains." *FLAIRS*.

Bangsø, O. and Wuillemin, P.H., 2000. *Object Oriented Bayesian Networks: A Framework for Topdown Specification of Large Bayesian Networks and Repetitive Structures*. Technical Report CIT-87.2-00-obphw1. Aalborg: Department of Computer Science, Aalborg University

Billingsley, P., 1995. *Probability and Measure*. New York, NY: Wiley.

Binford, T. and Levitt, T.S., 2003. "Evidential reasoning for object recognition." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(7), pp. 837-51.

Boutilier, C., N. Friedman, et al., 1996. Context Specific Independence in Bayesian Networks. *Uncertainty in Artificial Intelligence: Proceedings of the Twelfth Conference*, San Francisco, CA, Morgan Kaufmann.

Brachman, R.J., Fikes, R.E., and Levesque, H.J., 1983. "KRYPTON: A Functional Approach to Knowledge Representation." *IEEE Computer Society*, **16**(10), pp. 67-73.

Buntine, W.L., 1994. "Operations for Learning with Graphical Models." *Journal of Artificial Intelligence Research*, **2**, pp. 159-225.

Charniak, E. and Goldman, R.P., 1993. "A Bayesian Model of Plan Recognition." *Artificial Intelligence*, **64**, pp. 53-79.

Costa, P., 2005. *Bayesian Semantics for the Semantic Web*. Doctoral Dissertation, Fairfax, VA: School of Information Technology and Engineering, George Mason University. http://hdl.handle.net/1920/455.

Costa, P., Laskey, K.B., Fung, F., Pool, M., Takikawa, M., and Wright, E., 2005. "MEBN Logic: A Key Enabler for Network-Centric Warfare." *10th Annual Command and Control Research and Technology Symposium*.

Cowell, R.G., 1999. *Probabilistic Networks and Expert Systems*. Berlin: Springer-Verlag.

D'Ambrosio, B., 1991. "Local expression languages for probabilistic dependency." *Uncertainty in Artificial Intelligence: Proceedings of the Seventh Conference*, San Mateo, California, Morgan Kaufmann.

D'Ambrosio, B., Takikawa, M., Fitzgerald, J., Upper, D., and Mahoney, S.M., 2001. "Security situation assessment and response evaluation (SSARE)." *DARPA Information Survivability Conference & Exposition II*, IEEE Computer Society.

Davis, E., 1990. *Representations of Commonsense Knowledge*. San Mateo, California: Morgan Kaufmann.

de Finetti, B., 1934/1990. *Theory of Probability: A Critical Introductory Treatment*. 2nd English Edition, translated by A. Machi and A.F.M. Smith, New York: Wiley.

De Raedt, L. and Kersting, K., 2003. "Probabilistic Logic Learning." *ACM-SIGKDD Explorations: Special Issue on Multi-Relational Data Mining*, **5**(1), pp. 31-48.

DeGroot, M. H., 1982. "Lindley's Paradox: Comment " *Journal of the American Statistical Association* **77**(378), pp. 336-339.

DeGroot, M.H. and Schervish, M.J., 2002. *Probability and Statistics*. Boston, Massachusetts: Addison Wesley.

Elliott, R.J., Aggoun, L., and Moore, J.B., 1995. *Hidden Markov Models: Estimation and Control*. Berlin: Springer-Verlag.

Enderton, H.B., 2001. *A Mathematical Introduction to Logic*: Harcourt Academic Press.

Frege, G., 1879/1967. *Begriffsschrift*. Translated in J. Heijenoort (ed), *From Frege to Gödel*, Cambridge, MA: Harvard University Press.

Fung, F., Laskey, K.B., Pool, M., Takikawa, M., and Wright, E., 2005. "PLASMA: Combining Predicate Logic and Probability for Information Fusion and Decision Support." *AAAI Spring Symposium on Decision Support in a Changing World*.

Geiger, D. and Heckerman, D., 1991. "Advances in Probabilistic Reasoning." *Uncertainty in Artificial Intelligence: Proceedings of the Seventh Conference*, San Mateo, CA, Morgan Kaufmann Publishers.

Getoor, L., Friedman, N., Koller, D., and Pfeffer, A., 2001. "Learning Probabilistic Relational Models," in *Relational Data Mining*. Saso Dzeroski and Nada Lavrac (ed.), Berlin: Springer-Verlag.

Getoor, L., Koller, D., Taskar, B., and Friedman, N., 2000. "Learning Probabilistic Relational Models with Structural Uncertainty." *ICML-2000 Workshop on Attribute-Value and Relational Learning:Crossing the Boundaries*, Standford, California.

Ghahramani, Z., 1998. "Learning Dynamic Bayesian Networks," in *Adaptive Processing of Sequences and Data Structures: Lecture Notes in Artificial Intelligence*. C.L. Giles and M. Gori (eds.), Berlin: Springer-Verlag, pp. 168-97.

Gilks, W., Thomas, A., and Spiegelhalter, D.J., 1994. "A language and program for complex Bayesian modeling." *The Statistician*, **43**, pp. 169-78.

Glesner, S. and Koller, D., 1995. "Constructing Flexible Dynamic Belief Networks from First-Order Probabilistic Knowledge Bases." *ECSQARU*, pp. 217-26.

Grenander, U., 1996. *Elements of Pattern Theory*. Baltimore, MD: Johns Hopkins University Press.

Gruber, T.R., 1993. "A Translation Approach to Portable Ontology Specifications." *Knowledge Acquisition*, **5**(2), pp. 199-220.

Halpern, J.Y., 1991. "An Analysis of First-Order Logics of Probability." *Artificial Intelligence*, **46**(May), pp. 311-50.

Heckerman, D., Meek, C., and Koller, D., 2004. *Probabilistic Models for Relational Data*. MSR-TR-2004-30. Redmond, WA: Microsoft Corporation

Howson, C. and Urbach, P., 1993. *Scientific Reasoning: The Bayesian Approach*. Chicago, IL: Open Court.

IET, 2004. "Quddity*Suite Technical Guide." Arlington, VA: Information Extraction and Transport, Inc.

ISO/IEC, 2007. Information technology - Common Logic (CL) - A framework for a family of logic-based languages," ISO/IEC 24707:2007 - Geneva, Switzerland: International Organisation for Standardisation.

Jaeger, M., 1998. "Reasoning About Infinite Random Structures with Relational Bayesian Networks." *Proceedings of the 6th International Conference (KR '98)*.

Jaeger, M., 2001. "Complex Probabilistic Modeling with Recursive Relational Bayesian Networks." *Annals of Mathematics and Artificial Intelligence*, **32**, pp. 179-220.

Jaynes, E.T., 2003. *Probability Theory: The Logic of Science*. Cambridge, UK: Cambridge University Press.

Jensen, F.V., 2001. *Bayesian Networks and Decision Graphs*. Berlin: Springer-Verlag.

Kersting, K. and De Raedt, L., 2001. "Adaptive Bayesian Logic Programs." *Proceedings of the Eleventh International Conference on Inductive Logic Programming (ILP 2001)*, Springer-Verlag.

Koller, D. and Pfeffer, A., 1997. "Object-Oriented Bayesian Networks." *Uncertainty in Artificial Intelligence: Proceedings of the Thirteenth Conference*, San Francisco, CA, Morgan Kaufmann.

Langseth, H. and Nielsen, T., 2003. "Fusion of Domain Knowledge with Data for Structured Learning in Object-Oriented Domains." *Journal of Machine Learning Research*, **4**, pp. 339-68.

Laskey, K.B. and Costa, P., 2005. "Of Klingons and Starships: Bayesian Logic for the 23rd Century." *Uncertainty in Artificial Intelligence: Proceedings of the Twenty-first Conference*, Arlington, VA, AUAI Press.

Laskey, K.B., D'Ambrosio, B., Levitt, T.S., and Mahoney, S.M., 2000. "Limited Rationality in Action: Decision Support for Military Situation Assessment." *Minds and Machines*, Vol. 10: 53-77

Laskey, K.B. and Mahoney, S.M., 1997. "Network Fragments: Representing Knowledge for Constructing Probabilistic Models." *Uncertainty in Artificial Intelligence: Proceedings of the Thirteenth Conference*, San Mateo, CA, Morgan Kaufmann.

Laskey, K.B., Mahoney, S.M., and Wright, E., 2001. "Hypothesis Management in Situation-Specific Network Construction." *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference*, San Mateo, CA, Morgan Kaufman.

Lauritzen, S., 1996. *Graphical Models*. Oxford: Oxford Science Publications.

Levitt, T.S., Winter, C.L., Turner, C., J., Chestek, R.A., Ettinger, G.J., and Sayre, S.M., 1995. "Bayesian Inference-Based Fusion of Radar Imagery, Military Forces and Tactical Terrain Models in the Image Exploitation System/Balanced Technology Initiative." *International Journal of Human-Computer Studies*, **42**.

Lin, Y. and Druzdzel, M.J., 1997. "Computational Advantages of Relevance Reasoning in Bayesian Belief Networks." *Uncertainty in Artificial Intelligence: Proceedings of the Thirteenth Conference*, San Francisco, CA, Morgan Kaufmann.

Mahoney, S.M., 1999. *Network Fragments*. Faifax, VA: School of Information Technology and Engineering, George Mason University.

Mahoney, S.M. and Laskey, K.B., 1998. "Constructing Situation Specific Networks." *Uncertainty in Artificial Intelligence: Proceedings of the Fourteenth Conference*, San Mateo, CA, Morgan Kaufmann.

Mahoney, S.M. and Laskey, K.B., 1999. "Representing and Combining Partially Specified Conditional Probability Tables." *Uncertainty in Artificial Intelligence: Proceedings of the Fifteenth Conference*, San Mateo, CA, Morgan Kaufmann.

Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D.L., and Kolobov, A., 2005. "BLOG: Probabilistic Models with Unknown Objects." *Proceedings of the Nineteenth Joint Conference on Artificial Intelligence*.

Murphy, K., 1998. *Dynamic Bayesian Networks: Representation, Inference and Learning*. Berkeley, CA: Computer Science Division, University of California.

Natarajan, S., Tadepalli, P., Altendorf, E., Dietterich, T.G., Fern, A., and Restificar, A., 2005. "Learning First-Order Probabilistic Models with Combining Rules." *Proceedings of the 22nd International Conference on Machine Learning*.

Neapolitan, R.E., 2003. *Learning Bayesian Networks*. New York: Prentice Hall.

Ngo, L. and Haddawy, P., 1997. "Answering Queries from Context-Sensitive Probabilistic Knowledge Bases." *Theoretical Computer Science*, **171**, pp. 147-77.

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

Peirce, C.S., 1885. "On the Algebra of Logic." *American Journal of Mathematics*, **7**, pp. 180-202.

Pfeffer, A., 2000. *Probabilistic Reasoning for Complex Systems*. Stanford, CA, Stanford University.

Pfeffer, A., 2001. "IBAL: An Integrated Bayesian Agent Language." *Joint Conference on Artificial Intelligence (IJCAI)*.

Poole, D., 1993. "Probabilistic Horn Abduction and Bayesian Networks." *Artificial Intelligence*, **64**(1), pp. 81-129.

Poole, D., 2003. "First-Order Probabilistic Inference." *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*.

Russell, S. and Norvig, P., 2002. *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Prentice-Hall.

Sato, T., 1998. "Modeling Scientific Theories as PRISM Programs." *ECAI98 Workshop on Machine Discovery*.

Savage, L.J., 1954. *The Foundations of Statistics*. New York: Wiley.

Sowa, J.F., 2000. "Knowledge Representation: Logical, Philosophical and Computational Foundations," Brooks-Cole Publishers.

Spiegelhalter, D.J., Thomas, A., and Best, N., 1996. "Computation on Graphical Models." *Bayesian Statistics*, **5**, pp. 407-25.

Stone, L.D., Barlow, C.A., and Corwin, T.L., 1999. *Bayesian Multiple Target Tracking*. Boston, MA: Artech House.

Tarski, A., 1944. "The Semantical Concept of Truth and the Foundations of Semantics." *Philosophy and Phenomenological Research*, **4**.

Wellman, M.P., Breese, J.S., and Goldman, R.P., 1992. "From knowledge bases to decision models." *The Knoweldge Engineering Review*, **7**(1), pp. 35-53.

Whittaker, J., 1990. *Graphical Models in Applied Multivariate Statistics*. Chichester: John Wiley & Sons.