REASONING WITH MODELS OF PROBABILISTIC KNOWLEDGE OVER PROBABILISTIC KNOWLEDGE

BY

AFSANEH H. SHIRAZI

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

ABSTRACT

In multi-agent systems, the knowledge of agents about other agents' knowledge often plays a pivotal role in their decisions. In many applications, this knowledge involves uncertainty. This uncertainty may be about the state of the world or about the other agents' knowledge. In this thesis, we answer the question of how to model this probabilistic knowledge and reason about it efficiently.

Modal logics enable representation of knowledge and belief by explicit reference to classical logical formulas in addition to references to those formulas' truth values. Traditional modal logics (see e.g. [Fitting, 1993; Blackburn *et al.*, 2007]) cannot easily represent scenarios involving degrees of belief. Works that combine modal logics and probabilities apply the representation power of modal operators for representing beliefs over beliefs, and the representation power of probability for modeling graded beliefs. Most tractable approaches apply a single model that is either engineered or learned, and reasoning is done within that model.

Present model-based approaches of this kind are limited in that either their semantics is restricted to have all agents with a common prior on world states, or are resolving to reasoning algorithms that do not scale to large models.

In this thesis we provide the first sampling-based algorithms for model-based reasoning in such combinations of modal logics and probability. We examine a different point than examined before in the expressivity-tractability tradeoff for that combination, and examine both general models and also models which use Bayesian Networks to represent subjective probabilistic beliefs of agents. We provide exact inference algorithms for the two representations, together with correctness results, and show that they are faster than comparable previous ones when some structural conditions hold. We also present sampling-based algorithms, show that those converge under relaxed conditions and that they may not converge otherwise, demonstrate the methods on some examples, and examine the performance of our algorithms experimentally.

To my mom, Nina, and my brother, Omid, for their love and support.

ACKNOWLEDGMENTS

In writing this thesis I was helped and inspired by many colleagues and friends. It is an honor to thank them all for their support and encouragement, for the good times we have spent together and for being who they are. For those I have listed here, my gratitude goes way beyond the words I have assembled in their honor and for those I have missed, my love and appreciation is theirs nonetheless.

First and foremost, I would like to thank my advisor, Eyal Amir, for his endless guidance and support of my Ph.D. study and research, for his patience, motivation, enthusiasm, and immense knowledge. It has been an honor to be his first Ph.D. student. I appreciate all his contributions of time, ideas, and funding which made my Ph.D. experience productive and stimulating. The joy and enthusiasm he has for his research was contagious and motivational for me, even during tough times in my Ph.D. pursuit.

Besides my advisor, it is an honor for me to thank the rest of my thesis committee: Prof. Dan Roth, Prof. David Forsyth, and Prof. Chandra Chekuri, for their encouragement, insightful comments, and high expectations. I also would like to acknowledge Prof. Joe Halpern who was originally in my thesis committee for all the time he spent proof-reading my thesis and his priceless critiques. It is a pleasure to express my sincere gratitude towards our honorary thesis consultant, Dr. Brian Milch, who made this thesis possible with his enthusiasm, keen insight, valuable advice, and willingness to help in any way he could.

My sincere thanks also goes to Dr. Jianying Hu, Dr. Karolina Buchner, and

Dr. Ralph Rabbat, for offering me the summer internship opportunities in their groups and giving me the priceless opportunity to be a part of their diverse and exciting projects. I am indebted to my managers at Google, Bryan Horling and Othar Hansson, who kindly supported me in writing this thesis while working full-time on Google's innovative projects.

The members of the Knowledge Representation and Reasoning (KRR) research group have contributed immensely to my personal and professional experience at University of Illinois. The group has been a source of friendships as well as good advice and collaboration. I am very grateful to many members, past and present, of KRR research group including Deepack Ramachandran, Adam Vogel, Rodrigo de Salvo Braz, Megan Nance, Jaesik Choi, Abner Guzman-Rivera, and Mark Richards for their kind assistance and encouragement.

My time at University of Illinois was made enjoyable in large part due to the many friends and groups that became a part of my life. I am grateful for the time spent with roommates and friends, for my sports buddies and our memorable trips, for Afarin and Hamid's hospitality as I completed my degree, and for many other people and precious memories.

This thesis would not have been possible without the support and help of my friends in Champaign, Iran, and the Bay Area. Special thank goes to my best friends, Maryam Karimzadehgan and Afarin Jahangiri, for helping me get through the difficult times and for all the emotional support in the past four years. I also would like to express my sincere gratitude to Hossein Ahmadi and Saeed Maleki who both immensely helped me in proof-reading and writing this thesis. I would also like to express my thanks to the best roommate ever, Domenica Amato.

I owe my deepest gratitude to my dearest mom without whom writing this thesis would have been next to impossible. She stood by me throughout my graduate studies, supporting me spiritually and financially, believing in me, and loving me.

I would like to thank her for being the best mom one could ever imagine and for unconditionally supporting me every step of the way. Last but not least, I wish to thank my brother, Omid Shirazi, and my dad, Ali Shirazi, for their love and encouragement throughout my life.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The study of knowledge in artificial intelligence is both theoretical and applied. Answers to “what do we know?”, “what can be known?”, and “what does it mean to say that someone knows something?” apply to many important areas [Aumann, 1986; Fagin *et al.*, 1995]. Formal models of reasoning about knowledge use modal operators and logic to express *knowledge* and *belief*. These enable agents to take into account not only facts that are true about the world, but also the knowledge of other agents.

Reasoning about knowledge of other agents plays an important role in contexts ranging from conversations to imperfect-information games. For example, in a bargaining situation, the seller of a car must consider what the potential buyer knows about the car’s value. The buyer must also consider what the seller knows about what the buyer knows about the value, and so on. Another example is the card game of poker when a player may bluff (bet or raise with an inferior hand) to cause other players to believe she has a dominant hand, and thus they all fold.

1.1 Reasoning about the Knowledge of Agents

Traditional formal models of reasoning about knowledge use *modal logic* [Hintikka, 1962; Kripke, 1963] to express *knowledge* and *belief*. Modal logics enable representation of knowledge and belief by explicit reference to classical logical formulas in addition to references to those formulas’ truth values. They reify formulas with the aid of special language constructs called *modal operators*. Those modal operators, such as $\Box, \diamond, K, B$ in common applications, allow discussion in the logic about knowledge (e.g. by agents) of formulas and also allow relating such knowledge to properties that hold in the world. They enable discussion in the logic about nested knowledge and belief, including statements such as *player 1 believes that if player 2 has $100 then player 2 does not believe that player 1 has $100*, or *if a cure for cancer exists, then company A knows that it exists, but does not know what that cure is.*

In many applications, it is important to consider more graded forms of knowledge than complete knowledge or lack thereof (e.g. not true in any accessible state). For example, a car salesman may not know the buyer's estimate of the car's value, but may have a probability distribution over the buyer's estimate. Also, a poker player may not know for sure if his opponent is bluffing, but he may "believe so with some likelihood[1]" because he believes that the opponent has low cards and the opponent believes that the player does not think so.

Traditional modal logics (see e.g. [Fitting, 1993; Blackburn *et al.*, 2007]) cannot easily represent scenarios involving degrees of belief or probabilistic notions. This is most clearly seen in the Kripke semantics for modal logics. There, a modal operator is interpreted to refer to truth values across all accessible states or the existence of at least one state satisfying a formula. There is little room in the language or semantics of *propositional* modal logics for notions between *all* (*necessarily*) and at-least-one (*possibly*). One can apply first-order modal logics and explicit axiomatizations of probabilities, but those quickly become expensive computationally and sometimes undecidable (e.g. [Wolter, 1999]).

Probabilistic representations (e.g. [Pearl, 1988]) can represent graded degrees of belief in natural ways. They can represent beliefs over beliefs using distribu-

[1]We do not refer here to any formal statement but rather to an intuitive notion.

tions over distributions (e.g., Dirichlet priors over multinomial distributions [Blei *et al.*, 2003]). However, a standard Bayesian network does not naturally support modeling the beliefs of multiple agents (e.g. consider modeling poker in such a way). Furthermore, reasoning with these representations is computationally hard because they mix structure, different continuous variables, and sometimes also discrete variables.

Works that combine modal logics and probabilities [Fine, 1972; Fattorosi-Barnaba and Amati, 1987; Fagin and Halpern, 1994; Milch and Koller, 2000; Heifetz and Mongin, 2001; Ferreira *et al.*, 2008] combine the best of both worlds, applying the representation power of modal operators for representing beliefs over beliefs, and the representation power of probability for modeling gradual beliefs. These works enable representation of probabilistic beliefs over probabilistic beliefs using a language that combines probabilities with belief statements and a semantics that extends Kripke structures with probabilities.

Such systems are expressive, but they are not tractable for large general models. Presently, most works that combine modal logic with probabilities enable automatic reasoning based on axiomatizations of different languages and semantics. Reasoning from axioms (when that is possible) is many times expensive for large domains. A competing approach [Harsanyi, 1967; Milch and Koller, 2000], prefers to have a single model that is either engineered or learned, and reasoning is done within that model (sometimes extending the model as part of inference).

Present model-based approaches of this kind are limited in their semantics in that they allow only models in which all agents have a common prior on world states. There are elegant algorithms for inference in such models when the common prior is represented as a Bayesian Network of low treewidth, but this efficiency deteriorates when one wishes to simulate with such systems general different beliefs for different agents (which can be done to some degree using observations in those models). Finally, such models and inference are limited in scalability by the low-treewidth requirement.

1.2 The Technical Results in this Thesis

In this thesis we examine a different point than examined before in the expressivity-tractability tradeoff for the combination of modal logics and probability. Here, we examine general models and also models which use Bayesian Networks to represent subjective probabilistic beliefs of agents. We provide exact inference algorithms for the two representations, together with correctness and convergence results, show that they are faster than comparable previous ones when some structural conditions hold. We also present example applications, and examine their performance experimentally. Throughout, we compare with the closest related work. Towards the end of the thesis we put this work in broader perspective and also summarize the differences between this thesis and the closest related work.

We focus on a special case of models in the logic of knowledge and belief defined by Fagin and Halpern [Fagin and Halpern, 1994] that still permits different agents having different subjective beliefs. Our models are more expressive than those presented by Milch and Koller [Milch and Koller, 2000], but we keep our language more restricted than theirs in that we do not permit observations (observations can still be simulated in a restricted way by compilation into the model, and possible expansion of the model). Otherwise, our language is similar to the one presented in [Milch and Koller, 2000] and includes modal operators $Bel_a^{\leq r}$, $Bel_a^{\geq r}$, and $Bel_a^{=r}$. Formulas such as $Bel_a^{\leq r}(\psi)$ correspond to the intuition "agent a believes that the probability of ψ is at most r".

For these models we provide basic exact reasoning algorithms of two kinds – one more efficient with larger state spaces, and the other more efficient with shallow nesting of modal operators in queries (i.e. the largest number of modal operators in root-to-leaf paths in expression trees of queries is small). The first algorithm is a top-down recursive approach for evaluating query formulas in a model M and state s, taking time $O(l \cdot N^m)$, where N is the maximum number of *accessible* states (states whose subjective probability is greater than 0) from any state s', m is the *nesting height* of modal operators in the query (the largest number of modal operators in a path from any leaf of the expression tree of the query to its root), and l is the size of the query. These algorithms serve as a starting point for exploring the rest of the algorithms developed here.

A second exact algorithm works in a bottom-up fashion, and takes time $O(l \cdot |S|)$ for *probabilistic knowledge* (a special case in which subjective probabilities are the same in an equivalence class of states, and all states external to this class have subjective probability 0), and $O(|S|^2 \cdot l)$ for general models. Here, $|S|$ is the size of the state space.

We also introduce sampling-based algorithms for those models, and show that they converge to correct answers under some conditions on the query formula. We assume for those that for every subformula of the forms $Bel_a^{\leq r}(\psi)$ or $Bel_a^{\geq r}(\psi)$ of our query no state s' has the subjective probability of a being exactly r (i.e. $Bel_a^{=r}(\psi)$ does not hold in any state s'). We show that when this condition fails, the answers may not converge and sometimes are guaranteed (almost surely) not to converge. Otherwise, the algorithms converge in probability to the correct answer, and take time $O(l \cdot n^m)$, where n is the number of samples used per evaluation of a modal operator, and m, l as above. The exponential term is the result of sampling compounding with larger nesting heights.

A computationally attractive representation of these models uses Bayesian Network fragments (called Graphical Kripke Models (GKMs)) to represent agents' subjective probability distributions over the world states. We present one way of formulating such models, namely, a presentation in which all subjective probabilities of an agent are represented with a single Bayesian Network *fragment* (our Bayesian Network fragment does not include a prior over real-world states, and only gives such a distribution given that the agent is in a world state.)

Using Bayesian Network fragments to represent agents' subjective probability is an important contribution, in that it provides a factored representation of a Kripke model without requiring the common prior assumption of Milch and Koller [Milch and Koller, 2000] (MK). Instead, a GKM represents *state-specific* probability distributions using these Bayes Net fragments.

In fact, GKMs do more than lift the assumption that all agents share a common prior. They remove what MK call the "observation assumption", which asserts that each agent has a prior distribution and an agent's probability distribution in each state can be derived by conditioning that prior on some observations. GKMs do not require an agent's probability distributions in different equivalence classes of states to be related to each other in any way.

For this representation (GKMs) we provide an exact method and a sampling method for answering queries as well. Our exact method applies variable elimination for Bayesian Networks that are constructed during computation. The algorithm is based on our basic bottom-up approach mentioned above, and constructs solutions for modal subformulas from the bottom up. It also embeds a dynamic-programming computation in which we find solutions for many possible queries to the same Bayesian Network. As pointed out above, those Bayesian Networks represent in a factored form the subjective probabilities of agents. In the algorithm presented here, they are augmented to include queries of interest and the results of runs of the algorithm on subformulas of the present one.

Finally, we also introduce a sampling method that is tractable on larger models for which the exact method is intractable. This sampling method is based on our results for general models. It samples n states for every subformula of our query whose expression tree is rooted in a modal operator. For each sample it evaluates the subformula in a recursive fashion. The main difference between this one and the sampling approach for our first representation of models is that when we evaluate conditional probabilities we do so on the Bayesian Network, hence saving space while keeping computation time at a similar comparable $(|\Phi| \cdot l \cdot n^m)$, for Φ the number of random-variable symbols defining a domain.

1.3 Logic Useful in Reading this Thesis

In this section we review the framework of modal logic. This helps put in context some of the results that we report in this thesis.

Modal logic tries to capture qualified truths. Modalities in different logics include operators that are named *necessary, obligatory, true after an action, known, knowable, believed, provable, from now on, so far, since, and until* after the functionality that they intend to serve. Each one of these has its own truth semantics.

Since there are many modal logics, it is useful to specify a particular logic by the set of *valid* formulas in the language of modal logic.

The possible worlds semantics was introduced independently by [Kanger, 1957; Hintikka, 1962], and were widely spread after a paper by [Kripke, 1963].

Definition 1 A modal propositional logic is called *normal* if it meets the following conditions:

Prop (Rule: Propositional truths) Includes every tautology

K (Axioms: Kripke) $\Box(P \to Q) \to (\Box P \to \Box Q)$.

MP (Rule: Modus-Ponens) If it includes $X, X \to Y$ then it also includes Y.

N (Rule: Necessitation) If it includes X then it also includes $\Box X$.

Definition 2 A *Kripke model* is a triple $\mathcal{M} = \langle \mathcal{G}, \mathfrak{R}, v \rangle$, where $\mathcal{G}$ is the set of worlds in the model, $\mathfrak{R}$ is the accessibility relation, and v is the valuation of propositional letters. $\langle \mathcal{G}, \mathfrak{R} \rangle$ is called a *frame*, and intuitively represents a theory including the restrictions on $\mathfrak{R}$ and $\mathcal{G}$ (i.e., the restriction that $\mathcal{G}$ includes these worlds). We say that $\mathcal{M}$ is *based* on the frame $\langle \mathcal{G}, \mathfrak{R} \rangle$.

We write $\mathcal{M}, s \Vdash \varphi$ for φ *is true in the world s in the model $\mathcal{M}$.*

Definition 3 φ is *valid in a model $\mathcal{M}$* iff $\forall s \in \mathcal{M}\ \mathcal{M}, s \Vdash \varphi$. φ is *valid in a collection C of models* (said *φ is C-valid*) iff $\forall \mathcal{M} \in C\ \varphi$ is valid in $\mathcal{M}$. Given a frame F, it is treated as the collection of all models based on that frame.

The following conditions are used to define standard classes of modal logics for different applications.

K The class of all frames. This is the weakest normal modal logic.

$$K : \qquad \Box(P \to Q) \to (\Box P \to \Box Q)$$

T The class of all *reflexive* frames. The motivation is of $\Box$ as *knowable* or *necessary*.

$$T : \qquad \Box P \to P$$

Together with K, T characterizes **T**: $S \models_{\mathbf{T}} U \to X \iff S \cup [\![T]\!] \models_{\mathbf{K}} U \to X$ where $[\![T]\!]$ refers to the set of axioms T applied to every formula P in the language.

K4 The class of all *transitive* frames.

$$4 : \qquad \Box P \to \Box\Box P$$

Intuitively, $\Box$ refers to *known*, and it is also called *positive introspection*.

S4/KT4 The class of all *reflexive transitive* frames. $S \models_{\mathbf{S4}} U \to X \iff S \cup [\![T]\!] \cup [\![4]\!] \models_{\mathbf{K}} U \to X$.

KB The class of all *symmetric* frames.

$$B: \qquad P \to \Box \diamond P$$

$S \models_{\mathbf{KB}} U \to X \iff S \cup [\![B]\!] \models_{\mathbf{K}} U \to X$.

B/KBT The class of all *symmetric reflexive* frames. $S \models_{\mathbf{B}} U \to X \iff S \cup [\![B]\!] \cup [\![T]\!] \models_{\mathbf{K}} U \to X$.

S5/KT45/KT4B The class of all *symmetric, reflexive, transitive* frames.

$$5: \qquad \neg\Box P \to \Box\neg\Box P$$

This is *negative introspection*. $S \models_{\mathbf{S5}} U \to X \iff S \cup [\![T]\!] \cup [\![4]\!] \cup [\![B]\!] \models_{\mathbf{K}} U \to X \iff S \cup [\![T]\!] \cup [\![4]\!] \cup [\![5]\!] \models_{\mathbf{K}} U \to X$.

Read alone, 5 is the condition of *euclidity*: $\forall s, t, u\ R(s,t) \wedge R(s,u) \implies R(t,u)$.

D The class of all frames having *idealization* (i.e., every possible world has a possible world accessible from it)[2]. Intuitively $\Box$ is *obligatory*.

$$D: \qquad \Box P \to \diamond P$$

It is true iff $\diamond TRUE$.

D4 The class of *transitive idealized* frames.

[2]Also called *serialization*.

DB The class of *symmetric idealized* frames.

Representing modal logics in first-order logic (FOL) is rather simple for normal logics. The translation follows the principle of axiomatizing the restrictions on R in FOL. We transform every proposition in modal logic into a predicate, and follow recursively with the standard connectives. The only modification needed is: $\Box Z$ goes to $(\forall y)(R(x,y) \rightarrow Z(y))$ where x stands for the world that we are examining right now.

Theorem 4 X is **K**-valid iff $\forall x\ \tau(X,x)$ is valid (where τ is the translation mentioned above).

The translation to FOL from other modal logics can be carried by stating the restrictions on R as preconditions for the goal formula. For example, for $\vdash_{\mathbf{T}} X$ we write $(\forall x\ R(x,x)) \rightarrow \forall x\ \tau(X,x)$.

Modal logics have other semantics than the simple Frame/Relation/Possible-worlds semantics. We briefly outline them below.

Algebraic semantics uses a boolean algebra $\mathcal{A}$, and a mapping from $\mathcal{A}$ to $\mathcal{A}$ to give valuations of the $\Box$ operator.

General Frames are like Frames, but defining the propositions to be the set of worlds in which they hold. The set of propositions is then closed under the usual operators for sets.

Neighborhood frames semantics uses the General Frames idea, and instead of having a set of accessibles for every world, having the set of propositions $\mathcal{N}$, that are necessary in that world. We then define $\mathcal{M}, s \Vdash \Box X \iff \{s' | \mathcal{M}, s' \Vdash X\} \in \mathcal{N}(s)$. This semantics can express normal modal logics, but also others. They are sometimes called *minimal models*.

Epistemic Structures Generalize over the Neighborhood semantics, defining a possible world to be a pair of propositions and their valuations, rather than a syntactic object. We then get a kind of recursive definition. Possible worlds then fall naturally into levels of complexity.

Propositional **K**,**T**,**S4** are decidable and PSPACE-complete. Propositional **S5** is decidable and NP-complete.

1.4 Plan of this Thesis

The rest of the thesis is organized as follows. In Chapter 2 we provide the syntax and semantics that will be used throughout this thesis. This includes both our basic representation and the factored one using Bayesian Networks. Chapter 3 presents exact and approximate (sampling-based) reasoning algorithms for the basic representation, and also our approximation convergence and soundness of reasoning under the condition mentioned above. Chapter 4 presents exact and approximate (sampling-based) algorithms for our models' factored representation. Chapter 5 discusses related work, and we conclude with Chapter 7.

1.5 Publication Notes

Below is the list of my publications in chronological order and where they are used in this thesis:

- [Shirazi and Amir, 2005]
- [Shirazi and Amir, 2007]: Modified and improved in sections 2.1, 3.1, 3.2, and 3.3.

- [Shirazi and Amir, 2008]: Modified and improved in Sections 2.2, 3.5, 4.1 [3], and 4.4.
- [Hajishirzi *et al.*, 2009]
- [Shirazi *et al.*, 2009]
- [Shirazi and Amir, 2011]

[3]The algorithm appeared in the thesis only gets its idea from this paper, we improved it here and fixed part of it.

CHAPTER 2

AGENTS' PROBABILISTIC BELIEFS ABOUT OTHER AGENTS' BELIEFS

We define Probabilistic Belief about Belief Logic (PBBL) to be a special case of the logic of knowledge and belief defined by Fagin and Halpern [Fagin and Halpern, 1994] (hereforth FH). We adopt a restricted version of the syntax of the logic PEL [Milch and Koller, 2000] with the main difference being that we consider only non-conditioned belief formulas, and have no accessibility relation or observation sets. Our semantics is slightly more general than available there in that it allows agents to have different probability distributions at different states. We expand on the relationship between PBBL and PEL in Section 2.3.

2.1 Language and Semantics of PBBL

The language of PBBL is parameterized by a nonempty, finite set Φ of random-variable symbols, each with an associated finite domain, $dom(X)$; and a nonempty set $\mathcal{A}$ of agents.

Given these parameters, the language of PBBL consists of the following well-founded formulas (*formulas*):

- *atomic formulas* of the form $\top$, $\bot$, or $X = v$, where $X \in \Phi$ and $v \in dom(X)$ (the domain of X).
- *formulas* of the form $\neg\varphi$ and $\varphi \vee \psi$, where φ and ψ are PBBL formulas; we use $\varphi \wedge \psi$ as an abbreviation for $\neg(\neg\varphi \vee \neg\psi)$.

- *formulas* of the form $Bel_a^{\leq r}(\varphi)$, $Bel_a^{\geq r}(\varphi)$, and $Bel_a^{=r}(\varphi)$, where $a \in \mathcal{A}$, $r \in [0,1]$, and φ is a PBBL formula. (We refer to $Bel_a^{\leq r}, Bel_a^{\geq r}, Bel_a^{=r}$ as *modal operators*.)

This definition of well-founded formulas follows a standard line of defining well-founded formulas in logics at large (the interested reader can examine standard textbooks such as [Shoenfield, 1967]). Such constructions imply several properties that are worth pointing out: every formula has a finite length; every formula can be represented by a tree of connectives, modal operators, and atomic formulas, with the latter at the leaves and the former (connectives and modal operators) at the inner nodes of the tree; the set of (well-founded) formulas in a PBBL is of countable cardinality (assuming a countable number of r's are allowed in modal operators).

Our atomic formulas play the same role as propositions in the FH logic. The modal formula $Bel_a^{\leq r}(\varphi)$ (similarly, $Bel_a^{\geq r}(\varphi)$ and $Bel_a^{=r}(\varphi)$) should be read as "according to agent a, the probability of φ is at most r." (or "... at least r", or "exactly r", respectively).

In the following we proceed with definitions that include only $Bel_a^{\leq r}(\varphi)$ and omit the developments for $Bel_a^{\geq r}(\varphi)$ and $Bel_a^{=r}(\varphi)$, which are done in an identical manner. Nonetheless, in later chapters we point out subtle differences and careful considerations with reasoning about $Bel_a^{=r}(\varphi)$, but those are irrelevant until then.

We will provide formal semantics for these statements after defining a model theory for PBBL. Notice that our belief statements are more restricted than those available with PEL in that they do not contain observations. One may compile observations into distributions available in the models defined below. Also notice that the FH logic is more expressive in that it allows probabilities to be related by arbitrary linear inequalities.

Definition 5 A *Probabilistic Kripke Model* (*PKM*) M of the PBBL language having random variables Φ and agents $\mathcal{A}$ is a tuple $(S, \pi, \mathbf{P})$ in which

- S is a nonempty finite set of possible states of the world;
- π is a value function mapping each random-variable symbol $X \in \Phi$ to a discrete random variable X^M which is a function from S to $dom(X)$. Thus, the sample space of X^M is $dom(X)$, and every $s \in S$ determines a specific value x taken by X^M in s.
- $\mathbf{P} = \{P_a\}_{a \in \mathcal{A}}$, and every $P_a \in \mathbf{P}$ is a function from S to probability distributions over S (the *subjective probability distributions for agent* a)

Thus, every PKM specifies a set of states and maps each random-variable symbol to a random variable defined on those states. Regarding those, in the rest of the paper we sometimes refer to a random variable X^M simply as X, and the context would make it clear that X is not a random-variable symbol.

For $a \in \mathcal{A}$ and $s \in S$, $P_a(s)$ is a probability distribution over S. $P_a(s)(s')$ specifies the probability that agent a assigns to state s' when the agent is in fact in state s. In what follows, it is sometimes more convenient to write $P_{a,s}(s')$ for $P_a(s)(s')$. Thus, $P_{a,s}$ is agent a's subjective probability distribution at s. When A has only one agent a, we omit the agent's subscript from P_a and write P instead.

We say that s' *is accessible from* s according to agent a in PKM M, if $P_{a,s}(s') > 0$.

The semantics of PBBL is similar to ones for normal modal logic. We introduce a satisfaction relation $\models$ (aka *logical entailment*) such that $(M, s) \models \varphi$ means the formula φ is satisfied at world s in model M. We also define the set of states satisfying formula φ in model M as $[\varphi]_M = \{s \in S \mid (M, s) \models \varphi\}$.

Definition 6 For $\Phi, \mathcal{A}$ as above, a PKM $M = (S, \pi, \mathbf{P})$, state $s \in S$, and formula φ in PBBL, $(M, s) \models \varphi$ if one of the following holds:

- φ is an atomic formula $\top$.
- φ is an atomic formula $X = v$ and $X(s) = v$.
- $\varphi = \neg\psi$ and $(M, s) \not\models \psi$.
- $\varphi = \psi \vee \xi$ and $(M, s) \models \psi$ or $(M, s) \models \xi$.
- $\varphi = Bel_a^{\leq r}(\psi)$ and $P_{a,s}([\psi]_M) \leq r$.

There are applications such as MAIDs [Milch and Koller, 2003] in which one is interested in a satisfaction relation from models alone with an unknown state, e.g. $M \models \varphi$. MK include a prior probability over states in models of PEL to provide such satisfaction relations among others. In PBBL one can provide such a prior by introducing an external agent, a_0, into $\mathcal{A}$, and specifying queries from a_0's perspective. We show how to do so in more detail in Section 2.3.

There are philosophical and practical reasons to assume that each P_a adheres to some rules. For example, the agent can know that he is in an equivalence class of states purely by noticing his probability distribution and the value it assigns to every $s' \in S$. Therefore, it is useful to distinguish a class of PKMs that addresses such an intuition.

Definition 7 (Probabilistic Knowledge) A PKM $M = (S, \pi, \mathbf{P})$ is a *probabilistic knowledge model*, if for every $a \in \mathcal{A}$ there is an equivalence relation $R_a \subseteq S \times S$ such that

$$\forall s_1, s_2, s \in S \;\; P_{a,s_1}(s) = P_{a,s_2}(s) \text{ whenever } R(s_1, s_2)$$

and $P_{a,s}(s') = 0$ for every $s, s' \in S$ such that $\neg R_a(s, s')$. (We use the notation $\neg R_a(x, y)$ as a shorthand for $\langle x, y\rangle \notin R_a$.)

Thus, in a probabilistic knowledge model M every agent $a \in \mathcal{A}$ has an identical subjective distribution for every two states in the same equivalence class of R_a. This implies some properties reminiscent of the modal logic $S5$, such as the following (we bring this as an example; full comparison with $S5$ is outside the scope of this thesis).

Proposition 8 For M a probabilistic knowledge model, s a state in M, and φ a PBBL formula,

$$M, s \models Bel_a^{=1}(Bel_a^{=1}(\varphi)) \iff Bel_a^{=1}(\varphi)$$

PROOF First, we prove a lemma about the connective $\iff$, which holds in all PKMs (this holds as a result of our Definition 6, and also follows from the fact that our models are special cases the models of *Kripke Structures for Knowledge and Probability* of FH).

Lemma 9 Let M be a PKM, s a state in M, and φ, ψ formulas. Then $M, s \models \varphi \iff \psi$ iff ($M, s \models \varphi$ iff $M, s \models \psi$).

PROOF For the forward direction assume $M, s \models \varphi \iff \psi$. $\varphi \iff \psi$ is a shorthand for $(\varphi \wedge \psi) \vee (\neg\varphi \wedge \neg\psi)$, so the rules in Definition 6 apply as follows.

$M, s \models \varphi \iff \psi$ implies that $M, s \models \varphi \wedge \psi$ or $M, s \models \neg\varphi \wedge \neg\psi$. Without loss of generality assume the first case holds. Then $M, s \models \varphi$ and $M, s \models \psi$. The needed conclusion ($M, s \models \varphi$ iff $M, s \models \psi$) holds.

For the backward direction assume $M, s \models \varphi$ iff $M, s \models \psi$. Thus, if $M, s \models \varphi$ holds, then also $M, s \models \psi$ holds and hence (by Definition 6) also $M, s \models \varphi \wedge \psi$.

This implies $M, s \models \varphi \wedge \psi \vee (\neg\varphi \wedge \neg\psi)$ which is our wanted conclusion. A similar argument applies for the case that $M, s \not\models \varphi$. ■

The proof of Proposition 8 continues.

Let $\psi_1 = Bel_a^{=1}(Bel_a^{=1}(\varphi))$ and $\psi_2 = Bel_a^{=1}(\varphi)$. According to the lemma, it is enough to prove that $M, s \models \psi_1$ iff $M, s \models \psi_2$. Assume that ψ_1 holds in M, s ($M, s \models \psi_1$), and we shall prove that ψ_2 also holds in M, s.

Assume to the contrary. Then $M, s \not\models \psi_2$. This implies that $P_{a,s}([\varphi]_M) < 1$. Thus, there must be a state $s' \in S$ such that $R_a(s, s')$ and $M, s' \not\models \varphi$ and $P_{a,s}(s') > 0$.

By the definition of probabilistic knowledge models, $P_{a,s'}([Bel_a^{=1}(\varphi)]_M) = P_{a,s}([Bel_a^{=1}(\varphi)]_M) < 1$ for this s'. Thus, $M, s' \not\models Bel_a^{=1}(\varphi)$. Since $P_{a,s}(s') > 0$, this means that $P_{a,s}([Bel_a^{=1}(\varphi)]_M) < 1$. We conclude that $M, s \not\models \psi_1$, and our proposition is proved in the forward directions. The argument in the opposite direction is similar, only is carried in a reverse order. ■

For most of our development in this thesis (which focuses on inference) such assumptions and restrictions (e.g. of probabilistic knowledge) are welcome but not necessary. When we need such assumptions (e.g. to provide a faster algorithm for inference in Chapter 3), we point them out explicitly. The interested reader can look at [Fagin and Halpern, 1994] for a more detailed examination of conditions one could or should make on such models.

2.2 A Factored Representation for Probabilistic Kripke Models

A Bayesian Network (BN) is a pair $G, \mathbf{C}$, with $G = (V, E)$ being a directed acyclic graph in which nodes represent random variables, and $\mathbf{C} = \{C_X\}_{X \in V}$ are conditional probability distributions (CPDs), with C_X a conditional distribution over $dom(X)$ given values for X's parents in G. The joint distribution over variables in V is defined by $Pr(X_1, \ldots, X_m) = \prod_{i=1}^{m} Pr(X_i|\text{pa}(X_i))$, where $\text{pa}(X)$ denotes the set of variables which are parents of X in G.

Definition 10 A *Unified Representation* of a PKM $M = (S, \pi, \mathbf{P})$ of the PBBL language having random variables Φ and agents $\mathcal{A}$ is the tuple $(S, \pi, \mathbb{P})$ such that

- for every $a \in \mathcal{A}$ we create a fresh X^a (actual state) and X^h (hypothesized state) random variables defined on S; and
- $\mathbb{P} = \{\mathcal{P}_a\}_{a \in \mathcal{A}}$, and every $\mathcal{P}_a \in \mathbb{P}$ is a CPD over X^h given X^a such that

$$\mathcal{P}_a(X^h = s' \mid X^a = s) = P_{a,s}(s')$$

We use a Unified Representation of PKM M to capture for every agent a all the subjective probability distributions within a single probabilistic structure. X^a, X^h serve the technical purpose of denoting the actual state (X^a) and the possible state (X^h), allowing us to define $\mathcal{P}_a$ as a conditional probability.

In the following we write Φ^a for a set of fresh random-variable symbols $\{X_1^a, ..., X_n^a\}$ corresponding to the random-variable symbols in $\Phi = \{X_1, ..., X_n\}$, only with superscript a. We write Φ^h for a similarly fresh random-variable symbols set with superscripts h. For the reader familiar with Dynamic Bayesian Networks (DBNs), the following factored representation (product of conditional probability distributions here) will seem familiar and similar to the transition model used in DBNs.

Definition 11 (Factored Representation of PKMs) A *graphical Kripke model* (GKM) M of the PBBL language having random variables Φ and agents $\mathcal{A}$ is a tuple $(S, \pi, \mathbf{G}, \mathcal{C})$ such that

- $S = \prod_{X \in \Phi} dom(X)$.

- π is a value function mapping each random variable symbol $X_i \in \Phi$ to a random variable X_i^M as above, with the restriction that $X_i^M(s) = x_i$, for $s = \langle x_1, ..., x_m \rangle \in S$.

- $\mathbf{G} = \{G_a\}_{a \in \mathcal{A}}$, with G_a a directed acyclic graph (DAG) over vertices $\Phi^a \cup \Phi^h$ such that edges always arrive into Φ^h vertices (thus, Φ^a are always parents, and Φ^h are either parents or children or both in the DAG G_a).

- $\mathcal{C} = \{\mathbf{C}_a\}_{a \in \mathcal{A}}$, with $\mathbf{C}_a = \{C_{a,X^h}\}_{X^h \in \Phi^h}$ CPDs over $X^h \in \Phi^h$ given X^h's parents in G.

A GKM represents a Unified Representation of a PKM, with a factored representation (the product of CPDs) for the state-to-state CPDs in $\mathbb{P}$. As with BNs, factored representations assert conditional independence assumptions between variables and non-descendants given parent variables. Since our framework inherits all the relationships between factored models and their independence assumptions, we omit those relationships here, and only note that we inherit them from the theory of Bayesian Networks [Pearl, 1988].

Several insights are worth mentioning about GKMs: The graphical models provided by $\mathbf{G}, \mathcal{C}$ are different for each agent; Also, every PKM has an equivalent GKM (to see this, notice that every Unified Representation of a PKM can be represented with a GKM).

2.3 Comparing PBBL and PEL

In this section we compare our PBBL language with Milch and Koller's [Milch and Koller, 2000] PEL language.

One example that can be expressed in PEL of [Milch and Koller, 2000] and cannot straightforwardly be expressed in our PBBL is the formula $BelCond_a^{\leq r_1}($ $BelCond_a^{\leq r_2}(\varphi|\psi))$, i.e. the (unconditioned) beliefs of agent a about his beliefs once he observes ψ. Our PBBL language is more restricted than PEL in that PEL can represent observations of agents in the language. However, we can simulate those observations in a limited way by adding extra random-variable symbols as we show now.

In PEL conditioned beliefs are represented with the operator $BelCond_a^{\leq r}(\varphi|\psi)$. To simulate conditional belief operator $BelCond_a^{\leq r}(\varphi|\psi)$ we add a new agent, a_ψ, to the PBBL representation. The intuition here is that this agent is the same as agent a with the difference that this agent has observed formula ψ. To define this agent mathematically, we need to define $P_{a_\psi,s}$ as follows:

$$P_{a_\psi,s}(s') = \begin{cases} \frac{P_{a,s}(s')}{P_{a,s}([\psi]_M)} & \text{if } s' \in [\psi]_M; \\ 0 & \text{otherwise.} \end{cases}$$

when $s \in [\psi]_M$. However, when s is not in $[\psi]_M$, $P_{a_\psi,s}(s') = 1$ when $s = s'$ and 0 otherwise. Note that the second case is when it is not possible for agent a to observe ψ. Since PEL assumes that observations are not noisy it is impossible to observe that in s and therefore the equivalence class of state s would only include state s. Then $BelCond_a^{\leq r}(\varphi|\psi) = Bel_{a_\psi}^{\leq r}(\varphi \wedge \psi)$.

To see the effects of this simulation, observe the following example. Assume that there are 4 states of the world s_1, s_2, s_3, and $s4$. We have two boolean random variables X and Y in Φ. Let X be $true$ in s_1, s_2 and $false$ in s_3, s_4. Also, let Y

be $true$ in s_1, s_3 and $false$ in s_2, s_4. Let $P_{a,s}$ be defined as follows:

$$P_{a,s}(s') = \begin{cases} .4 & \text{if } s' \in \{s_1, s_4\}; \\ .1 & \text{otherwise.} \end{cases}$$

Now to simulate $BelCond_a^{\leq r}(Y = true|X = true)$ we add a new agent $a_{X=true}$. This agent stands for agent a when he observes that X is $true$ which means that agent a knows that he is either in s_1 or s_2. With the above explanation the new subjective probability distribution is defined as follows:

$$P_{a_{X=true},s}(s') = \begin{cases} \frac{.4}{.5} = .8 & s_1; \\ \frac{.1}{.5} = .2 & s_2; \end{cases}$$

when s is either s_1 and s_2. Then $BelCond_a^{\leq r}(Y = true|X = true) = Bel_{a_{X=true}}^{\leq r}($ $Y = true \wedge X = true)$ which is true when r is greater than or equal to $.8$.

Another difference between PEL and PBBL is that PBBL's models are less restricted than PEL's in that PBBL allows different (arbitrary) distributions for every agent at every state. PEL assumes that agents have a common prior probability distribution over states of the world, and an agent's local probability distribution at state s is equal to this global distribution conditioned on the set of states the agent considers possible at s.

For example, assume that there is a lamp and a switch and two agents. PEL assumes that there is a common prior probability distribution that the agents hold about the state of the world. Assume for example that given that the switch is on, the probability of the lamp being on is $.9$ and this probability is 0 given that the switch is off. Now if agent 1 has information that the power might be off in the area with some probability, he might have a different subjective probability distribution than that of agent 2. For example, according to agent 1's distribution, the probability of the lamp being on given that the switch is on might is $.4$. This second scenario can be handled with our model but not easily in PEL. We present a more detailed example towards the end of this section.

As a result of this difference, PBBL's usage of Bayesian Networks is in a different way (hence making different assumptions) than PEL. We factor the subjective probability models of every agent, whereas PEL factors the prior distribution.

Consequently, the BN fragments used here are different in their meaning and usage from the common-prior BNs used by MK. One implication of this is that if the number of equivalence classes (as described above) is large, the total size of the BN fragments in a GKM could be much larger than MK's single BN. The individual fragments may be simpler than a prior BN if the variables known to the agent induce useful independence properties; conversely, they may be more complex if the agent's observations add dependencies (e.g., between parents of an observed node).

A third difference between the two languages is that our PBBL models do not have a prior distribution over states where PEL does, but we can simulate this prior distribution by adding one extra agent to our language. With a prior probability distribution, [Milch and Koller, 2000] shows how to compute the probability of an arbitrary PEL formula without a specified state, i.e. defining and computing satisfaction queries of the form $M \models \varphi$.

To be able to define and compute the probability of a query of this form, we add an agent 0 to our model. Our intuition is that this agent corresponds to us, an outside observer of the system, or nature. We require $P_{0,s}(s')$ to satisfy

$$P_{0,s}(s') = P_{0,s''}(s') \text{ for all } s'' \in S$$

In other words, $P_{0,s}(s')$ is a prior distribution that nature has over S. Now $M, s \models Bel_0^{\leq r}(\phi)$ for any $s \in S$ is defined to be the meaning of $M \models \phi$. Computing whether such a query has value *true* is the same as comparing the probability of an arbitrary unconditioned PEL formula ϕ with r.

In Chapter 3, when we present reasoning methods for computing satisfaction of a PBBL formula ϕ, we show that we can also find an r' for which $Bel_a^{=r'}(\varphi)$ is true. To understand this r's relationship to PEL note that $(M, s) \models Bel_a^{=r}(\varphi)$ if and only if $P_{a,s}([\varphi]_M) = r$. $Bel_a^{=r}(\varphi)$ is an abbreviation for $Bel_a^{\leq r}(\varphi) \wedge Bel_a^{\leq 1-r}(\neg\varphi)$. The value of r is exactly the probability of an arbitrary unconditioned PEL formula.

Finally, below is an example of a PKM model that cannot be modeled with PEL unless we add an extra variable. Assume that $\Phi = \{X_1\}$, and there are two agents, a_1, a_2. Thus, there are two states, s_0, s_1, with $X_1 = 0$ in s_0 and $X_1 = 1$ in s_1. $dom(X_1) = \{0, 1\}$, and a GKM M specifies $P_{a_1,s_1}(X_1) = 0.2$ and $P_{a_2,s_1}(X_1) = 0.6$.

Representing this example in PEL we can distinguish between the agents by specifying different observation sets. An observation set for a_1 is either $\{\}$ (the empty set) or $\{X_1\}$, and the same goes for a_2. Thus, one of the agents, e.g. a_2, must have observation set $\{X_1\}$, and therefore has in PEL that $P_{a_2,s_1} = 1$. This does not agree with our GKM M. This shows that without further variables PEL cannot represent this M.

If one wishes to add variables to a model to represent different agents in PEL, one can do so at the price of creating a complex model as follows: we add one variable to the GKM that we call agent-identity, with the intuition that this variable switches between the agents. This variable then is added as a parent in the PEL BN to the rest of the variables, switching their CPTs from one agent to another.

A more difficult part to model, however, is the representation of agents subjective probabilities pertaining to each state. In our last example above, even a single agent GKM M as above cannot be represented in PEL without additional variables. To see this, let agent a_1 (our only agent here) have in M different probability over S when the agent is in s_1, s_2: $P_{a_1,s_2}(X_1) = 0.3$. Since the only observation variable possible is X_1, PEL cannot represent this scenario because one state must have the observation set $\{X_1\}$ in PEL. To represent this in PEL one may add a variable X_{11} to the model such that the probability of X_1 in the prior of PEL is dependent on X_{11}, which will act as a switching variable, i.e. $Pr_{PEL}(X_1|X_{11} = 0) = 0.2$ and $Pr_{PEL}(X_1|X_{11} = 1) = 0.3$.

Finally, observe that as the number of variables in our GKM M grows, representation with PEL becomes exponentially harder. For $\Phi = \{X_1, \ldots, X_n\}$ we have 2^n states in S. Let C_{a_1,X_j^h} in GKM M be such that X_j^h has a single parent $X^a_{1+[(j-2) \mod n]}$. Then, the switching variable mentioned above for a PEL model may need to have 2^n values, one for each state. Modeling this with n additional binary RVs instead does not solve the problem because some of the CPTs in the new PEL model will have n parents. Thus, such a representation with PEL in this way will take space $O(2^n)$, whereas a GKM would take space $O(n)$. It remains an open question whether any PEL that can represent the different subjective probabilities in this M would have an exponential-size representation.

2.4 Example

An application of this framework is modeling a Hold'em game (Texas Hold'em poker). In Hold'em, players receive two downcards as their personal hand, after which there is a round of betting. Three boardcards are turned simultaneously and another round of betting occurs. The next two boardcards are turned one at a time, with a round of betting after each. A player may use any five-card combination from the board and personal cards. Ranking rules of the game determine the winner.

Suppose that in a two player Hold'em game, the boardcards are $K\clubsuit K\heartsuit Q\heartsuit$ $3\spadesuit 2\clubsuit$, Player 1 has $A\diamondsuit K\diamondsuit$, and player 2 has $K\spadesuit 3\clubsuit$. From the perspective of player 1, player 2 can have any card except the boardcards and the cards in player 1's hand. In Hold'em, the possible worlds are all the possible ways the cards could have been distributed among the players. Initially, a player may consider possible all deals consistent with the cards in her hand with equal probability. Players may acquire additional information in the course of the game that allows them to eliminate some of the worlds they consider possible or change the probability distribution of the others.

We have 9 random-variable symbols in our Hold'em example, $\Phi = \{X_1, X_2, \ldots, X_9\}$ where X_1 and X_2 are player 1's cards, X_3 and X_4 are player 2's hand and the rest are the boardcards. $dom(X_i) = \{A\heartsuit, 2\heartsuit, \ldots K\heartsuit, \ldots, A\diamondsuit, \ldots, K\diamondsuit\}$.

In this example there are two players $\mathcal{A} = \{a, b\}$ (Alice and Bob, respectively). Figure 2.1 shows their perspective in a Hold'em game. $Bel_a^{\leq 0.2}((X_3 = A\heartsuit) \vee (X_4 = A\heartsuit))$ is an example of a formula in this language whose truth value can be evaluated on the current state of the world. This should be read as "according to Alice, the probability of Bob having $A\heartsuit$ (Ace of hearts) is at most 0.2".

In Hold'em, $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ is also an example of a formula. This query refers to "according to Bob, with high probability (at least 0.9) according to Alice the probability of Bob having Ace of hearts is low (at most 0.2)". Therefore, if Bob has Ace of hearts and it is the best possible hand, he can raise and Alice might call because she might assume he is bluffing.

Note that in our example we need a subjective probability distribution for each player. We model our Hold'em example with PKM $(S, \pi, \mathbf{P})$ in which S is the set of all possible states, that includes a state corresponding to each possible com-

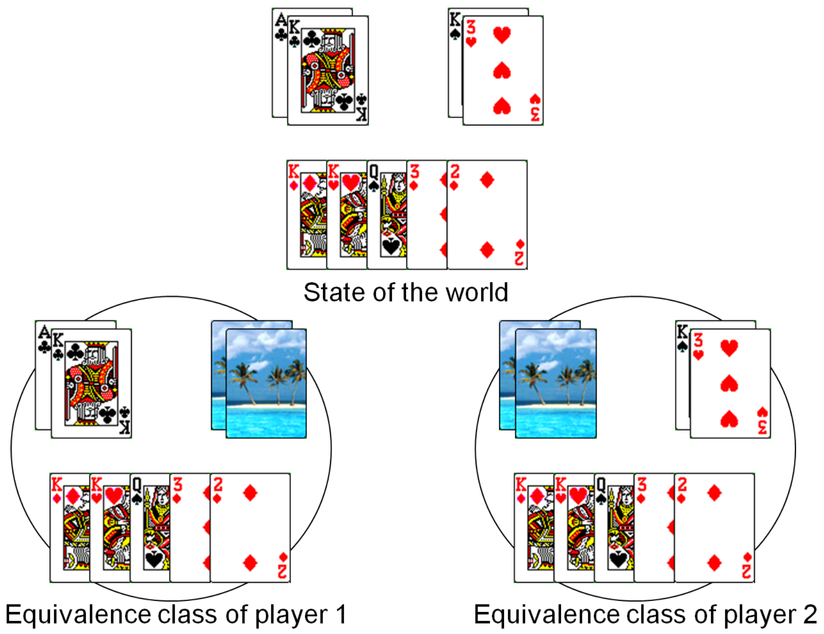

Figure 2.1: Alice and Bob perspective in Hold'em example.

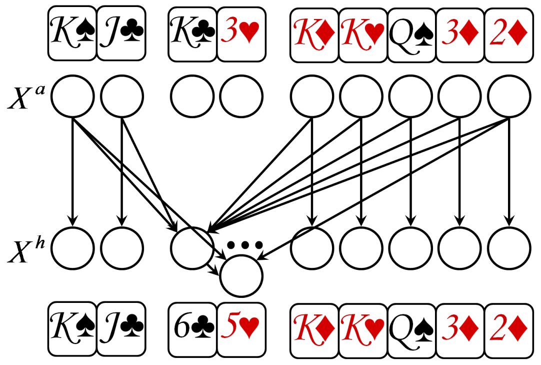

Figure 2.2: G_a of the GKM of Hold'em. Acyclic graph for Alice.

bination of player hands and boardcards. Since Alice knows the boardcards and her cards and does not know Bob's hand, $P_{a,s}(s')$ is positive when s is a state in which the value of random-variables are $A\diamondsuit K\diamondsuit$ K♠3♣ K♣$K\heartsuit Q\heartsuit$3♠2♣ and s' corresponds to $A\diamondsuit K\diamondsuit$ K♠3♣ K♣$K\heartsuit Q\heartsuit$3♠2♣. We assume that $P_{a,s}(s')$ is uniform on the states that are possible to a. This is the assumption that poker players make at the beginning of a game when the cards are dealt since there is only one card of each type in the deck.

Figure 2.2 shows G_a of the GKM of our Hold'em example. The nodes in the first row correspond to the actual state of the world, whereas the second row corresponds to a possible state of the world from the perspective of Alice. Each node takes values from $\{A\heartsuit, 2\heartsuit, \dots K\heartsuit, \dots, A\diamondsuit, \dots, K\diamondsuit\}$. The first and the second nodes are observed by player 1 to have values $K\spadesuit$, $J\clubsuit$, respectively. In each row, the first two nodes correspond to Alice's hand, the second two nodes correspond to Bob's hand, and the last five are the boardcards. From the perspective of Alice, Bob can have any cards except the boardcards and the cards in her hand. In G_a, this is shown by the edges to the third and forth nodes in X^h. The boardcards and player 1's hand cards are the same in the actual state of the world and the hypothetical state of the world.

$\mathbf{C}_a, \mathbf{C}_b$ are defined as follows:

$$C_{a,X_1^h}(X_1^h = v' | X_1^a = v) = \begin{cases} 1 & \text{if } v' = v; \\ 0 & \text{otherwise.} \end{cases} \text{ and the same for } X_2, X_5, \dots, X_9$$

$$C_{a,X_3^h}(X_3^h = v | X_1^a = v_1, X_2^a = v_2, X_5^a = v_5, \dots, X_9^a = v_9) = \frac{1}{\alpha}\begin{cases} 1 & \text{if } v \notin \{v_1, v_2, v_5, \dots, v_9\}; \\ 0 & \text{otherwise.} \end{cases}$$

As shown in the above equation, X_3^h given its parents has a uniform distribution. α is the normalization factor. The conditional probability function for X_4^h given its parents is similar to X_3^h except that X_4^h is a child of X_3^h and should not be equal to X_3^h as well.

$$C_{a,X_4^h}(X_4^h = v | X_1^a = v_1, X_2^a = v_2, X_3^h = v_3, X_5^a = v_5, \ldots, X_9^a = v_9) =$$
$$\frac{1}{\alpha}\begin{cases} 1 & \text{if } v \notin \{v_1, v_2, v_3, v_5, \ldots, v_9\}; \\ 0 & \text{otherwise.} \end{cases}$$

Note that this model is most useful when the size of the largest Conditional Probability Table (CPT) is much smaller than $|\mathcal{S}|^2$ or when the CPTs can be represented compactly (*e.g.,* uniform distribution). In those cases, the size of the GKM is much smaller than the size of the corresponding PKM ($O(|S|^2)$).

CHAPTER 3

REASONING WITH A PROBABILISTIC KRIPKE MODEL

In Chapter 2 we described the logic PBBL and a semantics for it. We described PKMs, models that capture some intuitions about agents reasoning about each others' probabilistic knowledge. Specifically, we defined PBBL formulas and what it means for a PBBL formula to hold in a PKM and a world state in that PKM.

In this chapter we are concerned with computational aspects of answering queries of the form $M, s \models \varphi$, where M is a PKM, s is a state in M, and φ is a PBBL formula (we keep the parameters of the language, Φ and $\mathcal{A}$, implicit in this discussion).

The following sections present two main methods for reasoning with PKMs: exact reasoning (a bottom-up approach and a top-down approach), and approximate reasoning by sampling. The exact reasoning methods are relatively straightforward, but require careful attention to ensure correctness of the method for our semantics. The sampling approach provides a reasoning method for large models or large queries with deeply nested formulas (many nested applications of modal operators in PBBL).

3.1 Exact Reasoning with a PKM: a Top-Down Approach

In this section we provide exact reasoning algorithms for answering queries about state s in PKM M. We focus on satisfaction queries of the form $M, s \models \varphi$, for φ a formula in PBBL.

Figure 3.1 presents Function Top-Down (ToDo), a top-down approach to answering our satisfaction queries. It works as follows.

We represent a query formula φ with an expression tree [Kozen, 1997]. In an expression tree, the root is the query itself and the leaves are propositional primitives. Function Top-Down (ToDo) of Figure 3.1 starts from the root of the query's expression tree and recursively computes the value of its subformulas on a given state. The running time of the function grows exponentially with the number of nested modal operators in our query.

Theorem 12 (Soundness of ToDo) For PKM M, state s in M, and formula φ in PBBL, ToDo(M,s,φ) always terminates and returns a value in $\{true, false\}$. It returns a value *true* iff $M, s \models \varphi$.

PROOF The definition of formulas of PBBL implies that every formula corresponds to one of the 6 cases listed in ToDo (lines 1 through 6). The program terminates because (a) φ is a formula and thus of finite length; (b) every one of the recursive-call steps 4-6 decreases the size of the formula sent down the recursion; (c) once the recursive call reaches atomic formulas (steps 1-3), these steps always return with a value in $\{true, false\}$. Finally, the returned value from steps 4-6 is also in $\{true, false\}$, and we conclude the first part of the theorem (the program terminates and returns a value in $\{true, false\}$.

We now prove the second part of the theorem, namely, the soundness of ToDo with respect to our semantics. We prove this by induction on the structure of the

```
FUNCTION ToDo(PKM $M$, State $s$, Query $\varphi$)

  1. if $\varphi = \top$ then return true
  2. if $\varphi = \bot$ then return false
  3. if $\varphi$ is an atomic formula $X = v$, then return true if $X(s) = v$ and false
     otherwise.
  4. if $\varphi = \neg\psi$ then return true if ToDo($M$, $s$, $\psi$) return false, and return
     false otherwise.
  5. if $\varphi = \psi \vee \xi$ then return true if one of ToDo($M$, $s$, $\psi$) or ToDo($M$, $s$, $\xi$)
     returns true; return false otherwise.
  6. $\varphi$ is of the form $Bel_a^{\leq r}(\psi)$ [respectively, $Bel_a^{\geq r}(\psi)$, $Bel_a^{=r}(\psi)$]. Do the
     following:
     (a)  $prob \leftarrow 0$
     (b)  for all states $s' \in S$ such that $P_{a,s}(s') > 0$ do
     (c)     if ToDo($M$, $s'$, $\psi$) returns true, then $prob \leftarrow prob + P_{a,s}(s')$
     (d)  return true if $prob \leq r$ [respectively, $prob \geq r$, $prob = r$]; Other-
          wise, return false
```

Figure 3.1: A Top-Down (ToDo) reasoning algorithm for answering satisfaction queries of the form $M, s \models \varphi$ with PKM M, state s, and PBBL formula φ.

formula φ.

In the base of our induction φ is an atomic formula. Assume that $M, s \models \varphi$. If $\varphi = \top$, then Step 1 returns *true*. φ cannot be $\bot$ because Definition 6 implies $M, s \not\models \bot$. Finally, φ can be of the form $X = v$, in which case $M, s \models \varphi$ implies that $X(s) = v$, so ToDo will return *true* at Step 3. A similar argument shows the base of our induction for $M, s \not\models \varphi$.

For the induction step, the connectives $\neg$ and $\vee$ are treated by Steps 4,5 which correspond precisely to the definition of semantics for $\varphi = \neg\psi$ and $\varphi = \psi \vee \xi$.

The proof for $\neg$ is the following. $\varphi = \neg\psi$ for some ψ. Assume $M, s \models \varphi$. Then, $M, s \not\models \psi$. From our induction hypothesis, ToDo(M, s, ψ) must return *false*. So, Step 4 of ToDo(M, s, φ) will return *true*, and our induction step is complete for this case. The case of $M, s \not\models \psi$ is argued in a similar way.

The induction step for $\vee$ is similar to $\neg$. We develop in more detail the argument about the modal operators, focusing on $\varphi = Bel_a^{\leq r}(\psi)$ (the induction step for the other modal operators follows in the same way).

Assume that $M, s \models \varphi$. Then, $P_{a,s}([\psi]_M) \leq r$. The loop in Step 6 (a-c) sums up in $prob$ the probabilities $P_{a,s}(s')$ of all s' such that $P_{a,s}(s') > 0$ and ToDo(M,s',ψ) returns *true*. From our induction hypothesis, ToDo(M,s',ψ) returns *true* iff $M, s' \models \psi$. $M, s' \models \psi$ iff $s' \in [\psi]_M$, so

$$prob = \sum_{s' \in [\psi]_M} P_{a,s}(s') = P_{a,s}([\psi]_M).$$

From the definition of semantics of PKM, $M, s \models \varphi$ implies that $prob \leq r$, so Step 6(d) returns *true*, concluding our induction step. A similar argument holds for the case of $M, s \not\models \varphi$ and the other modal operators.

The converse follows the same argument in reverse and our proof is done. ■

Function ToDo traverses a recursion tree, with recursion steps occurring at Steps 4-6. The *nesting height* of modal operators in φ is the largest number of modal operators in a path from any leaf of the expression tree of φ to its root. This nesting height of modal operators dominates the computational cost of computing logical entailment with PKM M, state s, and formula φ, as we show in the following proposition.

The length of φ, denoted $|\varphi|$, is the total number of atomic-formula instances, modal operator instances, and connective instances in φ.

Proposition 13 Let φ be a query whose value on state s in PKM M we want to compute. Let $l = |\varphi|$, let m be the nesting height of modal operators in φ, and let n be the maximum number of states accessible from any state. Function

ToDo(M,s,φ) terminates in time $O(l \cdot n^m)$.

PROOF We prove the proposition by induction on the structure of φ, showing that $Time(l, n, m) \leq Cln^m$ for some constant C that is the time taken for 5 (an arbitrary number that is large enough for our purposes) basic instructions in our algorithm.

For $l = 1$, φ is an atomic formula (and also $m = 0$), in which case ToDo returns in time $\leq C = C \cdot 1 \cdot 1$.

For $l > 1$ and $\varphi = \psi \vee \xi$, Step 5 is taken for φ, and

$$\begin{aligned} Time(l, n, m) &\leq Time(|\psi|, n, m) + Time(|\xi|, n, m) \\ &\leq C|\psi|n^m + C|\xi|n^m + 1 \\ &\leq C \cdot l \cdot n^m \end{aligned}$$

A similar argument holds for $\varphi = \neg\psi$.

For $\varphi = Bel_a^{\leq r}(\psi)$,

$$\begin{aligned} Time(l, n, m) &\leq n \cdot (Time(|\psi|, n, m-1) + 1) \\ &\leq n \cdot C(l-1) \cdot n^{(m-1)} + 1 \\ &\leq C \cdot l \cdot n^m \end{aligned}$$

Our induction step is complete, and the proposition holds. ■

For queries $\varphi = Bel_a^{=r}(\psi)$ we can also find the value of r that $M, s \models Bel_a^{=r}(\psi)$. To do that we can change our algorithm slightly. In addition to returning the truth value of a formula we also return a number between 0 and 1. We only return this number when our query is $\varphi = Bel_a^{=r}(\psi)$. This number is the r that satisfies this formula. To return that, we change line 6(d) of Function ToDo to return $prob$ if

$\varphi = Bel_a^{=r}(\psi)$. This $prob$ is the sought out r. It is useful when a player wants to know the actual probability of the set of states that satisfy ψ.

3.2 Exact Reasoning with a Probabilistic Knowledge PKM: a Bottom-Up Approach

In Function ToDo, duplicate computations arise when the same state s' is visited from many different states s, and a subformula is evaluated in s' each time. We can overcome this inefficiency, if we take a bottom-up approach. There, we start from the bottom of the query's expression tree and avoid computing a subformula multiple times.

Function Knowledge-Bottom-Up (KBU) is described in Figure 3.2 and computes the value of a satisfaction query with a probabilistic knowledge PKM M while avoiding some recomputations.

Function KBU finds the answer to a satisfaction query given Probabilistic Knowledge PKM M, the equivalence relations for M (together with a precomputed set of equivalence classes), and a state s in M. It computes the value of an innermost modal-operator-rooted subformula of φ (with no nested modal function in it) for all equivalence classes, and associates the result with all the states in their respective equivalence classes.

We implement this precomputation and caching (line 2(c) in Function KBU) with a new random-variable symbol and a new random variable mapping every state $s \in S$ to $\{true, false\}$. We replace the modal-operator-rooted subformula ξ in φ with this new random-variable symbol. We continue this replacement in a bottom-up fashion from the bottom of the formula's expression tree to its top, replacing subformulas with new random-variable symbols.

From Proposition 13 we know that ToDo(M,s,φ) returns in time $O(|\varphi|)$, so

```
FUNCTION KBU(Probabilistic Knowledge PKM $M$, $\mathbf{R} = \{R_a\}_{a \in \mathcal{A}}$ equivalence relations for $M$, state $s$, formula $\varphi$)
  1. $m \leftarrow 1$
  2. while there is a modal operator in $\varphi$, do
     (a) Select $\xi$ a subformula rooted in a modal operator, $\xi = Bel_a^{\leq r}(\psi)$ (or
         the other modal operators $Bel_a^{\leq r}$ or $Bel_a^{=r}$) such that $\psi$ has no modal
         operator
     (b) Let $X_m$ be a fresh random-variable symbol, and extend $\pi$ in $M$ to in-
         terpret $X_m$ as a random variable with $dom(X_m) = \{true, false\}$ (a
         function from $S$ to $\{true, false\}$ – we define its values on $S$ below)
     (c) for every $e$ equivalence class of $R_a$ do
         i. $t \leftarrow \left(\sum_{s' \in e, ToDo(M,s',\psi)=true} P_{a,s'}(s')\right)$
         ii. for every $s' \in e$ define $X_m(s') = true$ if $t \leq r$ and false other-
             wise
     (d) replace $\xi$ in $\varphi$ with $X_m = true$
     (e) $m \leftarrow m + 1$
  3. return ToDo($M$, $s$, $\varphi$)
```

Figure 3.2: Bottom-Up reasoning algorithm about satisfaction queries with Probabilistic Knowledge PKMs

each call for Function ToDo in Lines 2(c)i and 3 takes linear time. Computation in KBU becomes more expensive with every run of 2(c), which calls Function ToDo $O(|S|)$ times. That part replaces modal operators with new random variables. Thus, in Function KBU, we visit each state once for each modal operator instance in φ.

Theorem 14 (Soundness and Time of KBU) For Probabilistic Knowledge PKM M, equivalence relations $\mathbf{R}$ on M, state s in M, and formula φ in PBBL, KBU(M, $\mathbf{R}$, s, φ) always terminates in time $O(|\varphi| \cdot |S|)$ and returns a value in $\{true, false\}$. It returns a value *true* iff $M, s \models \varphi$.

PROOF To show that KBU always terminates with a return value in $\{true,$

$false\}$ first notice that every step of the while loop (Step 2) terminates. This is because steps 2(a),2(b) are simple linear operations in $|\varphi|$; and 2(c) takes $O(|S|)$ calls to Function ToDo, each one on a formula without modal operators, thus taking time $O(|\psi|)$, where ψ is the subformula being replaced by a new variable in this stage of the iteration.

The while loop terminates because the size of φ decreases with each iteration by at least $|\psi|$ because we replace the subformula ξ (with length $|\xi| = |\psi|+1$) with an atomic formula with length 1. Let Sel be the set of all ψ such that $\xi = Bel_a^{\leq r}(\psi)$ is selected at Step 2(a) of the algorithm. Then, the time taken by the loop is $O(\sum_{\psi \in Sel} |\psi| \cdot |S|) = O(|\varphi| \cdot |S|)$. This proves the first part of the theorem.

To prove soundness of KBU, first look at the case $M, s \models \varphi$, and we prove that KBU(M,$\mathbf{R}$,s,φ) returns *true*. We prove this by induction on the number m of modal operators in φ.

For $m = 0$, the algorithm returns $ToDo(M, s, \varphi)$, so *true* (by Theorem 12).

For the induction step, $m > 0$. Let ξ be the first subformula chosen at step 2(a) of KBU. In the same loop iteration, let φ_0,M_0 be the original inputs to KBU, and let φ_1, M_1 be the resulting φ,$\mathcal{M}$ at Step 2(d). We show that $M_1, s \models \varphi_1$ (we already assume $M_0, s \models \varphi_0$), and the induction hypothesis will ensure that the rest of the run in KBU will return *true*.

Assume to the contrary that $M_1, s \not\models \varphi_1$. Notice that φ_1, φ_0 differ only on ξ being replaced by $X_m = true$. From the recursive construction in Definition 6 (the semantics of any PKM) it must be that that at some s_1 $X_m^{M_1}(s_1) = true$ and $P_{a,s_1}([\psi]_{M_0}) \not\leq r$ or $X_m^{M_1}(s_1) \neq true$ and $P_{a,s_1}([\psi]_{M_0}) \leq r$.

Let s_1 be such a state, and assume the first case ($X_m^{M_1}(s_1) = true$ and $P_{a,s_1}^{M_0}($ $[\psi]_{M_0}) \not\leq r$). From the way we constructed X_m in steps 2(b),2(c) we get that $t \leq r$ at step 2(c)ii. Let e be the equivalence class of s_1 (notice that equivalence classes of P^{M_0} are still equivalence classes of P^{M_1}). Then,

$$\begin{aligned}
&P^{M_0}_{a,s_1}([\psi]_{M_0}) = \\
&\textstyle\sum_{s' \in S, (M_0,s') \models \psi} P^{M_0}_{a,s_1}(s') = \\
&\textstyle\sum_{s' \in S, ToDo(M_0,s',\psi)=true} P^{M_0}_{a,s_1}(s') = \\
&\textstyle\sum_{s' \in S, ToDo(M_0,s_1,\psi)=true} P^{M_0}_{a,s_1}(s') = \\
&\textstyle\sum_{s' \in e, ToDo(M_0,s',\psi)=true} P^{M_0}_{a,s'}(s') \leq r
\end{aligned}$$

(notice that the replacement of s' by s_1 in the 3rd equality is possible because ψ has no modal operators. The replacement of s_1 with s' in the 4th equality is warranted by a similar argument and noticing that we restrict the summation now to $s' \in e$.)

Thus, $P^{M_0}_{a,s_1}([\psi]_{M_0}) \leq r$, contradicting our assumption of the first case. Therefore, we proved the induction hypothesis for the first case. The symmetric case of $X^{M_1}_m(s_1) \neq true$ is argued in a similar way, and so is the case of $M_0, s \not\models \varphi_0$, so our proof is done. ■

Compared to the runtime of function ToDo which is exponential in the number of nested modal operators, the runtime of KBU is linear in the length of the query (i.e. total number of modal functions). As a drawback, KBU is linear in the total number of states. When the number of nested modal operators is small, function ToDo is faster than KBU. To choose the best method for our application, we compare n^m with $|S|$ and decide whether to use ToDo or KBU.

3.3 Exact Reasoning with a general PKM: a Bottom-Up Approach

The bottom-up approach can be extended to general PKMs. The General-Bottom-Up (GBU) algorithm for reasoning in PKMs without the Probabilistic Knowledge

```
FUNCTION GBU(PKM $M$, state $s$, query $\varphi$)
  1. $m \leftarrow 1$
  2. while there is $\xi = Bel_a^{\leq r}(\psi)$ subformula of $\varphi$ where $\psi$ has no modal
     operator do
     (a) Add a fresh random-variable symbol $X_m$ to $\Phi$ with $dom(X_m) =$
         $\{true, false\}$, and map $\pi(X_m)$ to a fresh random variable $X_m^M$ tak-
         ing value in $dom(X_m)$ for every $s \in S$ (we set those values in Step
         2(c) below).
     (b) for every $s_2 \in S$, set $todo_{s_2} \leftarrow$ ToDo($M$,$s_2$,$\psi$)
     (c) for every $s_1 \in S$ do
         i. Set $t_{true}^{s_1} \leftarrow \sum_{s_2 \in S, todo_{s_2}=true} P_{a,s_1}(s_2)$;
         ii. set $X_m(s)$ to true if $t_{true}^{s_1} \leq r$; and to false otherwise
     (d) Replace $\xi$ in $\varphi$ with $X_m = true$
     (e) $m \leftarrow m + 1$
  3. return $ToDo(M, s, \varphi)$
```

Figure 3.3: General-Bottom-Up (GBU) reasoning algorithm for satisfaction queries $M, s \models \varphi$ with PKM M, state s, and formula φ.

assumption is shown in Figure 3.2.

Theorem 15 (Soundness and Time of GBU) For PKM M, state s in M, and formula φ in PBBL, GBU(M,s,φ) always terminates in time $O(|S|^2 \cdot |\varphi|)$ and returns a value in $\{true, false\}$. It returns a value *true* iff $M, s \models \varphi$.

PROOF To show that GBU always terminates with a return value in $\{true, false\}$ first notice that every step of the while loop (Step 2) terminates. This is because steps 2(a),2(d),2(e) are simple linear operations in $|\varphi|$; and Step 2(b) takes $O(|S|)$ computations of Function ToDo (each one on a formula without modal operators, thus taking time $O(|\psi|)$, where ψ is the subformula being replaced by a new variable in this stage of the iteration) and step 2(c)i takes $O(|S|^2)$. Thus, the time for every iteration of the loop is $O(|S| \cdot (|S| + |\psi|))$.

The while loop terminates because the size of φ decreases with each iteration by at least $|\psi|$ because we replace the subformula ξ (with length $|\xi| = |\psi| + 1$) with an atomic formula with length 1. Let Sel be the set of all ψ such that $\xi = Bel_a^{\leq r}(\psi)$ is selected at Step 2 of the algorithm. Then, the time taken by the loop is $O(\sum_{\psi \in Sel}(|S|^2 + |\psi| \cdot |S|)) = O(|S|^2 \cdot |\varphi|)$. This proves the first part of the theorem.

To prove soundness of GBU, first look at the case $M, s \models \varphi$, and we prove that GBU(M,$\mathbf{R}$,s,φ) returns *true*. We prove this by induction on the number m of modal operators in φ.

For $m = 0$, the algorithm returns $ToDo(M, s, \varphi)$, so *true* (by Theorem 12).

For the induction step, $m > 0$. Let ξ be the first subformula chosen at step 2 of GBU. In the same loop iteration, let φ_0,M_0 be the original inputs to GBU, and let φ_1, M_1 be the resulting φ,$\mathcal{M}$ at Step 2(d). We show that $M_1, s \models \varphi_1$ (we already assume $M_0, s \models \varphi_0$), and the induction hypothesis will ensure that the rest of the run in GBU will return *true*.

Assume to the contrary that $M_1, s \not\models \varphi_1$. Notice that φ_1, φ_0 differ only on ξ being replaced by $X_m = true$. From the recursive construction in Definition 6 (the semantics of any PKM) it must be that that at some s_1 $X_m^{M_1}(s_1) = true$ and $P_{a,s_1}([\psi]_{M_0}) \not\leq r$ or $X_m^{M_1}(s_1) \neq true$ and $P_{a,s_1}([\psi]_{M_0}) \leq r$.

Let s_1 be such a state, and assume the first case ($X_m^{M_1}(s_1) = true$ and $P_{a,s_1}^{M_0}($ $[\psi]_{M_0}) \not\leq r$). From the way we constructed X_m in steps 2(a),2(b),2(c) we get that $t_{true}^{s_1} \leq r$ at step 2(c)ii. Notice that $t_{true}^{s_1} = P_{a,s_1}^{M_0}([\psi]_{M_0})$ by the definition of $[\psi]_M$.

Thus, $P_{a,s_1}^{M_0}([\psi]_{M_0}) \leq r$, contradicting our assumption of the first case. Therefore, we proved the induction hypothesis for the first case. The symmetric case of $X_m^{M_1}(s_1) \neq true$ is argued in a similar way, and so is the case of $M_0, s \not\models \varphi_0$, so our proof is done. ■

Although the bottom-up algorithms are faster than our top-down method when the number of nested modal operators grows, bottom-up methods are still costly when the number of states is large.

3.4 Sampling Subgraphs

In this section we provide a reasoning method that uses sampling to answer queries on probabilistic knowledge structures. As we showed in the previous section, the running time of the bottom-up reasoning methods depend on the number of states. Sometimes this number of states can be very large, such as when the state space is defined by the cross product of random-variable domains in a large Bayesian Network. In those cases when the number of accessible states from every state is large, the approaches that we described above are intractable.

The bottleneck of those approaches, making them intractable for such domains, is the need for evaluation of terms on all states (or all accessible states) in the domain. This need is related to the evaluation of formulas rooted with a modal operator. If the number of states accessible from state s in PKM M is too large, evaluating $M, s \models Bel_a^{\leq r}(\psi)$ on that state needs to evaluate $M, s' \models \psi$ on all s' accessible from s.

Figure 3.4 presents Function ApxToDo, which samples from the states in S to estimate $M, s \models Bel_a^{\leq r}(\psi)$ and $M, s \models Bel_a^{\geq r}(\psi)$. It samples uniformly from that set, and assumes that such a uniform sampler is given to us. Also, we assume that the query formula φ includes no modal operators of the form $Bel_a^{=r}$, the reason being convergence rather than computational – our sampler will almost always (in a statistical sense) converge to *false* for a subformula rooted in such a modal operator for all r besides possibly 0 or 1. For the propositional connectives $\vee$ and $\neg$ and the atomic formulas ApxToDo behaves in an identical manner to Function

```
FUNCTION ApxToDo(PKM $M$, State $s$, Query $\varphi$) $\varphi$ a formula in PBBL without the modal operator $Bel_a^{=r}$

  1. if $\varphi = \top$ then return true
  2. if $\varphi = \bot$ then return false
  3. if $\varphi$ is an atomic formula $X = v$, then return true if $X(s) = v$ and false otherwise.
  4. if $\varphi = \neg\psi$ then return true if ApxToDo($M$, $s$, $\psi$) return false, and return false otherwise.
  5. if $\varphi = \psi \vee \xi$ then return true if one of ApxToDo($M$, $s$, $\psi$) or ApxToDo($M$, $s$, $\xi$) returns true; return false otherwise.
  6. $\varphi$ is of the form $Bel_a^{\leq r}(\psi)$; [respectively, $Bel_a^{\geq r}(\psi)$, but not $Bel_a^{=r}(\psi)$].
     Set $prob \leftarrow 0$
  7. for all $i \in \{1, ..., n\}$ ($n$ is the number of sample states that we wish to generate and use per modal operator)
     (a) Select $s' \in S$ an independently drawn sample from a uniform distribution over $S$.
     (b) if ApxToDo($M$, $s'$, $\psi$) returns true, then $prob \leftarrow prob + P_{a,s}(s')$
  8. return true if $prob \cdot |S|/n \leq r$; [respectively, $prob \geq r$]; Otherwise, return false
```

Figure 3.4: Sampling-based Top-Down (ApxToDo) reasoning algorithm for answering satisfaction queries of the form $M, s \models \varphi$ with PKM M, state s, and PBBL formula φ.

Todo.

Function ApxToDo traverses a recursion tree, with recursion steps occurring at Steps 4-7. Similar to Function ToDo, the nesting height of modal operators dominates the computational cost of computing logical entailment with PKM M, state s, and formula φ, as we show in the following proposition. The main difference here is that the base of the exponential term is now n, the number of samples we choose for evaluating Step 7 in ApxToDo (compared with $|S|$ in Step 6 of ToDo).

Proposition 16 Let φ be a query whose truth value on state s in PKM M we want to compute. Let $l = |\varphi|$, let m be the nesting height of modal operators in φ, and let n be the *number of sampled states* chosen in Step 7. Function ToDo(M,s,φ) terminates in time $O(l \cdot n^m)$.

PROOF The present proof follows the proof of Proposition 13, only replacing n with $n =$ the number of sampled states at each visit to Step 7. ■

Recall that a series $\{X_i\}_i \in \mathcal{N}$ of random variables *converges in probability* towards X if for all $\epsilon > 0$

$$\lim_{n \to \infty} Pr(|X_n - X| \geq \epsilon) = 0$$

In the following theorem we state convergence in probability to truth values $true$ or $false$. In fact, what we mean is that when the numerical values $1, 0$ replace $true, false$, then convergence holds as is stated in the definition of the convergence in probability.

Theorem 17 (Soundness of ApxToDo) For PKM M, state s in M, and formula φ in PBBL, ApxToDo(M,s,φ) always terminates and returns a value in $\{true$, $false\}$. Assume further that all modal operators in φ are of the form $Bel_a^{\leq r}$ or $Bel_a^{\geq r}$, and that for every subformula $\xi = Bel_a^{\leq r}(\psi)$ of φ, $M, s' \not\models Bel_a^{=r}(\psi)$ (NOTICE: "$=$" intended here).

Then, if $M, s \models \varphi$, its returned value converges to *true* in probability (taking *true* to be the numerical value 1 and *false* to be the numerical value 0) as $n \to \infty$. Similarly, If $M, s \not\models \varphi$, its returned value converges to *false* in probability as $n \to \infty$.

Before we prove this theorem notice that any subformula of the form $Bel_a^{=r}(\psi)$ can be simulated by $Bel_a^{\leq r}(\psi) \wedge Bel_a^{\geq r}$). We assume that there is no such subformula of φ because the only value it may converge on is *false*.

PROOF The first part of the theorem follows in a similar way to the proof of Theorem 12.

We now prove the second part of the theorem, namely, the convergence of ToDo to the correct semantics. We prove this by induction on the structure of the formula φ. For this, we prove that for every PBBL formula φ there is a monotonically decreasing function $f_\varphi(n)$ such that $f_\varphi(n) \to 0$ as $n \to \infty$ and $Prob(ApxToDo(M, s, \varphi) \neq ToDo(M, s, \varphi)) \leq f_\varphi(n)$.

In the base of our induction φ is an atomic formula. No samples are considered and the proof follows in an identical manner to that of Theorem 12. Here, $f_\varphi(n) = 0$.

For the induction step, the connectives $\neg$ and $\vee$ are treated by Steps 4,5 which correspond precisely to the definition of semantics for $\varphi = \neg\psi$ and $\varphi = \psi \vee \xi$.

The proof for $\neg$ is the following. $\varphi = \neg\psi$ for some ψ. Assume $M, s \models \varphi$. Then, $M, s \not\models \psi$. From our induction hypothesis, ToDo(M, s, ψ) returns *false* with probability $\geq 1 - f_\psi(n)$, for $f_\psi(n)$ as above. So, Step 4 of ToDo(M, s, φ) will return *true* in those times, and our induction step is complete for this case. The case of $M, s \not\models \psi$ is argued in a similar way.

For Step 5, $\varphi = \psi \vee \xi$. Assume $M, s \models \varphi$. Then, $M, s \models \psi$ or $M, s \models \xi$. Without loss of generality, assume the former. From our induction hypothesis, ToDo(M, s, ψ) returns *true* with probability $\geq 1 - f_\psi(n)$, for $f_\psi(n)$ as above. So, Step 5 of ToDo(M, s, φ) will return *true* in those times, and our induction step is complete for this case, with $f_\varphi(.) = f_\psi(.)$.

Now, assume the second case, namely, that $M, s \not\models \varphi$. Then, $M, s \not\models \psi$ and $M, s \not\models \xi$. From our induction hypothesis, ToDo(M, s, ψ) returns *false* with probability $\geq 1 - f_\psi(n)$, for $f_\psi(n)$ as above. Similarly, ToDo(M, s, ξ) returns *false* with probability $\geq 1 - f_\xi(n)$, for $f_\xi(n)$ as above.

Step 5 of ToDo(M, s, φ) will return *true* in those times in which both recursive calls return *false*. Take $f_\varphi(n) = 1 - (1 - f_\psi(n)) \cdot (1 - f_\xi(n))$, i.e. the probability of failure in our ApxToDo being the (independent) failure for either of ψ or ξ (or both). Thus, our induction step is complete for this case.

Finally, we regard the modal operators, focusing on $\varphi = Bel_a^{\leq r}(\psi)$ (the induction step for $Bel^{\geq r}$ follows in the same way).

Assume that $M, s \models \varphi$. Then, $P_{a,s}([\psi]_M) \leq r$. The loop in Step 7 (a-b) sums up in $prob$ the probabilities $P_{a,s}(s')$ of all sampled s' such that $P_{a,s}(s') > 0$ and ApxToDo(M,s',ψ) returns *true*.

We consider the number of times ApxToDo(M,s',ψ) returns an incorrect answer. It returns *true* with probability $\geq 1 - f_\psi(n)$ whenever $M, s' \models \psi$. Also, it returns *false* with probability $\geq 1 - f_\psi(n)$ whenever $M, s' \not\models \psi$. Thus, the probability of error is $\leq f_\psi(n)$ for each sample s'.

Let $prob_e = \sum_{s' \in [\psi]_M} P_{a,s}(s') = P_{a,s}([\psi]_M)$, the exact summation which we are trying to approximate with $prob$ in Step 7. Let $\epsilon = r - prob_e$. From our assumption in the theorem statement, $\epsilon > 0$.

Notice first that our induction hypothesis implies that the mean of ApxToDo(M, s', ψ) approaches the mean $\mu = prob_e/|S|$ as $n \to \infty$ (again, we take ToDo(M, s', ψ)$=$ $true$ to have value 1 and otherwise value 0). Thus, also $prob/n$ approaches this mean (mean of expectations).

Importantly, our induction hypothesis implies that the convergence of each sample (viewed as a random variable) to its expected value is in probability. Using the weak law of large numbers we get convergence in probability of the mean of our samples to the mean's expectation, i.e.,

$$\lim_{n \to \infty} Pr\left(\left|\frac{prob}{n} - \frac{prob_e}{|S|}\right| < \frac{\epsilon}{|S|}\right) = 1$$

Thus, also the convergence of $prob \cdot |S|/n$ to $prob_e$ is in probability (and reaches closer to $prob_e$ than ϵ distance), and there is a function $f_\varphi(.)$ as needed by our induction step.

A similar argument holds for the case of $M, s \not\models \varphi$ and the other modal operator, and our proof is done. ■

3.5 Properties of PKMs: Non-Convergence of Sampling

Our theorem above left open cases in which one could suspect that sampling may not converge. In this section we show that the estimated values of some queries almost surely does not converge at all when the number of samples increases. Consequently, the only way to answer these queries is to use an exact method. The following theorem states this result.

Let $\varphi(s)$ stand for 1 when $M, s \models \varphi$ and 0 otherwise.

Theorem 18 Let $Bel_a^{\leq r}(\varphi)$ be a query, s be a state, and $s_1, s_2, \ldots$ be a sequence of independent and identically distributed states sampled from $P_{a,s}(s')$. Define $\hat{Bel}_m = \frac{\varphi(s_1)+\ldots+\varphi(s_m)}{m}$ to be the observed value of $P_{a,s}([\varphi]_M)$ using m samples.

$$Pr(\lim_{m\to\infty}(\hat{Bel}_m \leq r) \text{ does not exist }) = 1$$

when $r = P_{a,s}([\varphi]_M) = \frac{1}{2}$.

PROOF Let X_m denote the random variable defined by

$$X_m = \begin{cases} 1 & \hat{Bel}_m \leq r \\ 0 & \text{otherwise} \end{cases}$$

Thus, we wish to show $Pr(\lim\limits_{m\to\infty} X_m \text{ does not exist }) = 1$

First, recall that to show $\lim_{m\to\infty} X_m$ does not exist it is enough to show that $\forall N > 0$ there are $n_1, n_2 \geq N$ such that $X_{n_1} = 1$ and $X_{n_2} = 0$. Thus, to show $Pr(\lim_{m\to\infty} X_m$ does not exist $) = 1$ it is enough to show that

$$Pr(\forall N > 0\ \exists n_1, n_2 \geq N\ X_{n_1} = 1, X_{n_2} = 0) = 1$$

We apply a known solution for the Gambler's Ruin (Ruin for short) problem for generalized one-dimensional random walks [Feller, 1968] (page 366). The Ruin problem is stated as follows.

At each step a particle has finite domain of motions, in which with probability p_k the particle moves from any point x to $x + k$, where the k may be zero, positive, or negative. Let X^{step} designate the random step taken by this particle. Hence, $Pr(X^{step} = k) = p_k$.

Given $a > 0$ (the gambler's goal), the Ruin problem is the problem of finding the probability u_z^a, such that starting from a position z such that $0 < z < a$ the particle will arrive at some position ≤ 0 before reaching any position $\geq a$.

Theorem 19 ([Feller, 1968], page 366) The solution of the Ruin problem satisfies the following inequality if X^{step} has zero mean. Let μ^+ and ν^- be defined, respectively, as the largest and the smallest k for which $p_k \neq 0$. (Thus, $Pr(X^{step} = \mu^+) > 0$ and $Pr(X^{step} = \nu^-) > 0$.) Then,

$$\frac{a - z}{a + \nu^- - 1} \leq u_z^a \leq \frac{a + \mu^+ - z - 1}{a + \mu^+ - 1}$$

We map our problem on the Ruin problem as follows. We have a particle that at each step moves either $\mu^+ = 1 - r$ with probability r or $\nu^- = -(r)$ with probability $1 - r$. This problem has mean zero because $r\mu^+ = -(1 - r)\nu^-$, so we can use Theorem 19.

The probability that the particle reaches the origin from position z before going over a is u_z^a. By Theorem 19

$$\lim_{a\to\infty} Pr(\text{reaching the origin before going over } a) = \lim_{a\to\infty} u_z^a = 1$$

Now, recall that we are trying to prove

$$Pr(\forall N > 0\ \exists n_1, n_2 \geq N\ X_{n_1} = 1, X_{n_2} = 0) = 1$$

Let ν_n^- be the number of ν^- occuring until and including step n. Let $z_n = (\nu_n^- \cdot \nu^-) + (\mu_n^+ \cdot \mu^+)$.

Notice that $z_n > 0$ iff $X_n = 0$. To see this, observe that $z_n = (\nu_n^- \cdot \nu^-) + (\mu_n^+ \cdot \mu^+) = (1-r)\mu_n^+ - r\nu_n^- = (1-r)\mu_n^+ - r(n - \mu_n^+) = \mu_n^+ - rn = n\hat{Bel}_n - rn$. Then for $X_n = 0$ we have $\hat{Bel}_n > r$, so $z_n = n\hat{Bel}_n - rn > 0$.

Let $N > 0$, and we find (in probability) $n \geq N$ such that $X_n = 1$. If $z_N \leq 0$, take $n = N$, and we have found $X_n = 1$ as needed.

Otherwise, $z_N > 0$ and we find $n > N$ as needed in the following.

$Pr(X_n = 0 \text{ for all } n > N) = Pr(z_n > 0 \text{ for all } n > N)$.

We notice that z_n reaching position ≤ 0 for $n > N$ for any a implies that "$z_n \geq 0$ for all $n > N$" does not hold for the sequence.

Let $\epsilon > 0$. Since $\lim_{a\to\infty} u_z^a = 1$ for $z = z_N$, there is a such that $u_z^a \geq 1 - \epsilon$.

$Pr(z_n > 0 \text{ for all } n > N) =$

$1 - Pr(z_n \leq 0 \text{ for at least one } n > N) \leq$

$1 - Pr(z_n \leq 0 \text{ for at least one } n > N,\ n < n' \text{ and } n' > N \text{ first such that } z_{n'} \geq a) =$

$1 - u_z^a \leq$

$1 - (1 - \epsilon) = \epsilon$

Thus $Pr(z_n > 0 \text{ for all } n > N) \leq \epsilon$ for every $\epsilon > 0$. Thus, $Pr(z_n > 0 \text{ for all } n > N) = 0$.

Thus,

$$Pr(\forall N > 0\, \exists n \geq N\, X_n = 1) = 1$$

A similar proof shows the opposite side, namely,

$$Pr(\forall N > 0 \exists n\, X_n = 0) = 1$$

This concludes our proof. ■

3.6 Example

In our Texas Holdem example of Chapter 2, $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ is an example of a query formula. If Alice and Bob are players a and b respectively, this query refers to Bob's belief about Alice's belief. This query refers to "according to Bob, with probability at least 0.9 Alice's probability of Bob having Ace of Hearts is low (at most 0.2)".

Computing the truth value of a query on a PKM is straightforward in theory. Given the PKM, answering a satisfaction query with no modal operator is done in $O(\text{the length of the query})$. The expensive part of computing the truth value of a satisfaction query is the part with modal operator. Assume that the number of possible states with probability greater than zero from a state is n (i.e, the number of s' that $P_{a,s}(s') > 0$). For query $Bel_a^{\geq 0.9}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$, the number of states on which the value of $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ should be calculated is n. Therefore if we have m nested belief operators we might need to calculate formulas on $O(n^m)$ states.

We explain ToDo with an example. Assume that we want to compute the truth

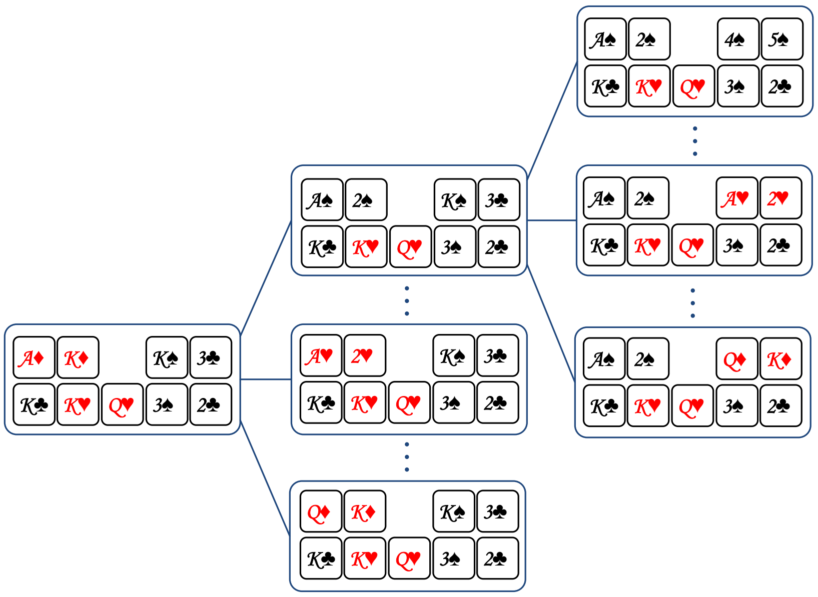

Figure 3.5: Alice and Bob perspective in Holdem example.

value of $\varphi = Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ on PKM of Section 2.4. Assume that s is $A\diamondsuit K\diamondsuit\ \ K\spadesuit 3\clubsuit\ \ K\clubsuit K\heartsuit Q\heartsuit 3\spadesuit 2\clubsuit$. We want to answer if $M, s \models Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ holds or not. In this example φ is $Bel_b^{\geq 0.9}(\psi)$ for $\psi = Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$, and so we need to compute the truth value of ψ on all states such that $P_{b,s}(s') > 0$. This probability is the subjective probability of Bob. Since Bob knows his cards and boardcards, the only states that he considers possible with positive probability are those that correspond to different cards for Alice which is shown in the middle column of Figure 3.5.

As shown in Figure 3.5, we need to compute $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ on all the states in the middle column. Now for each of these states we recursively compute the value of $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$. We show all the states accessible from one of them in the rightmost column of Figure 3.5. Note that this time $P_{a,s'}(s)$ should be greater than 0 which is the subjective probability of Alice.

The value of $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ is computed on the states of rightmost column. We add $P_{a,s}$ of the states in this column for which the value of $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ is true. Since the probability distribution is uniform, we can just count those states and divide the number by total number of states. This number is then compared with 0.2. Here, there are 44 states in which X_3 is $A\heartsuit$ (because 8 cards are known and 44 cards are remained in the deck) and there are 44 states in which X^4 is $A\heartsuit$. This means that 88 out of 45×44 states evaluates to true. This probability adds up to $\frac{88}{45\times 44} = 0.02$ which is smaller than 0.2. Therefore the value of $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ on state $A\spadesuit 2\spadesuit\ K\spadesuit 3\clubsuit\ K\clubsuit K\heartsuit Q\heartsuit 3\spadesuit 2\clubsuit$ is true.

Similarly, we compute satisfaction query $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ on all of the other middle column states. Then we use these values to compute the value of $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ which is true in this example.

In function ToDo subformulas of a query might be computed several times. We can overcome this inefficiency if we take a bottom-up approach instead. There, we start from the bottom of the query's expression tree and avoid recomputing a subformula. Functions KBU and GBU are designed for this purpose.

In KBU, we compute the value of all innermost modal operators (with no nested modal operator in them) for all equivalence classes, and associate the results with all the states in their respective equivalence classes. In function KBU, we visit each state once for each subformula.

To compute our example explained above, we first compute $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ on all states in S. Then for each equivalence class of Alice we compute $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$. An equivalence class of Alice is shown in right column of Figure 3.5. We associate the value computed for each equivalence class to all the states in that class. Next, we use these values to compute $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ on equivalence classes of Bob (one of them shown in the middle column of Figure 3.5).

Although the bottom-up approaches are faster than the top-down method for queries with large height of nested modal operators, it is slower when the height is small and the number of states is large.

We also provide a reasoning method that uses sampling to answer queries on PKMs. As we showed above, the running time of KBU and GBU depends on the number of states (linear and quadratic, respectively).

In our Holdem example, the number of states is 52^9. However, if both players have observed the boardcards we can define a PKM with a smaller state space. We assume that the domain of each of the boardcards has only one value in it, which is the actual value observed by both players and the domain of the players' cards has the 47 remaining cards in it. In that case, the number of states is equal to 47^4 which makes the bottom-up approach tractable. However, in a two player game with hundreds of cards when each player has tens of cards, it is not practical to use a bottom-up approach that computes the value of each subformula on all the states of the world.

The other option is to use ToDo. However, all the states accessible from a state should be visited to evaluate a satisfaction query. If the number of states accessible from a state is too large, evaluating $Bel_a^{\geq 0.9}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ on that state would be expensive. To avoid this expensive computation, Alice (based on her hand) only samples a few possible hands of Bob and computes the probability of transitioning to those hands and based on that gets an estimate of the actual value of $Bel_a^{\geq 0.9}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$. We proved that this value converges to the actual value when the number of samples grows.

We explain ApxToDo on the same example. Let the satisfaction query be $\varphi = Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ and s be $A\diamondsuit K\diamondsuit \quad K\spadesuit 3\clubsuit$ $K\clubsuit K\heartsuit Q\heartsuit 3\spadesuit 2\clubsuit$. Again, we want to answer if $M, s \models Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ holds or not.

As shown in Figure 3.5, in an exact method we need to compute $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ on all the states in the middle column. However, we sample only n states from the middle column and compute our subformula only on those states. Now for each of these states we recursively compute the value of $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$. Let's assume that the top state of middle column is one of our samples. Now, instead of computing $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ on all rightmost states, we again only sample n of those states. Then to compute the truth value of $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ we sum $P_{a,s'}(s)$ for the states in which $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ is true and multiply by $\frac{|S|}{n}$. Assume that $n = 10$ and in none of the samples Bob has $A\heartsuit$, so we compare $0 \times \frac{|S|}{10}$ with 0.2 and return true.

Similarly, we compute satisfaction query $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ on other sampled states of the middle column. Then we use these values to compute the value of $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$.

CHAPTER 4

REASONING WITH A FACTORED GRAPHICAL KRIPKE MODEL

In this chapter we provide reasoning methods for answering queries over GKMs. Recall that in Chapter 2 we defined GKMs to be factored representations of PKMs. A GKM includes a factored representation akin to a set of Bayesian Networks of the subjective probabilities held by agents at different states, representing all the subjective probabilities of any one agent with a single product of conditional distributions arranged in a DAG structure.

Our previous chapter showed that using GKMs potentially reduces the size of the model exponentially. The reasoning methods given in Chapter 3 for PKMs can be used on GKMs. However, these algorithms apply explicit enumeration of all states in the PKM, so applying them for GKMs may take exponential amount of time in the size of the model, rendering their usage for models of many variables impractical.

In this chapter we present algorithms for reasoning with GKMs that take advantage of GKMs' factored representations for faster inference. We will show that these algorithms are more efficient than their counterparts for PKMs. In Sections 4.1 and 4.2 we provide exact and sampling algorithms for answering PBBL queries of the form $M, s \models \varphi$, for φ a formula in PBBL.

4.1 Ordered Variable Elimination for GKMs

In this section we provide an algorithm to answer PBBL queries of the form $M, s \models \varphi$. The algorithm that we introduce has a main subroutine called *Ordered Variable Elimination for GKMs* (GKMOVE), and the overall algorithm is called sGKMOVE, for *GKMOVE value on state s*. Function sGKMOVE and its main subroutine, GKMOVE, are shown in Figure 4.1.

Subroutine GKMOVE is given a GKM M and a formula φ, and computes a table with columns corresponding to a subset of Φ (the way this subset is determined will be explained shortly) and a final column with values in $\{true, false\}$. Note that Φ is the set of random-variable symbols. Let that subset of Φ be $\Gamma = \{X_1, ..., X_u\}$. Then, every row in the table includes at column $i \leq u$ a value in $dom(X_i)$, and at column $u + 1$ a value in $\{true, false\}$. The table is complete in the sense that it has exactly one row for every $\langle x_1, ..., x_u \rangle \in \prod_{i=1}^{u} dom(X_i)$.

sGKMOVE takes the table computed by GKMOVE, finds the (unique) row $\langle x_1, ..., x_u, t \rangle$ such that $X_1(s) = x_1, ..., X_u(s) = x_u$, and returns t.

GKMOVE works by recursion and subsequent variable elimination (VE) in Bayesian Networks generated during computation. It always returns a structure of the form $\langle \Gamma, T \rangle$, where Γ is a set of random-variable symbols from Φ and T is either *true* or *false* (in which case $\Gamma = \emptyset$), or a table as described above.

At a high level, GKMOVE examines the expression-tree structure of φ, and recursively calls GKMOVE with subtrees of that expression tree. Whenever it encounters an atomic subformula or one whose expression tree is rooted in a propositional connective, it returns an easy-to-compute value (for atomic subformulas) or the result of the corresponding logical operation on tables and variable sets returned from recursive calls with subformulas of φ.

The interesting computation occurs when the GKMOVE receives (in the process of recursive computation) φ which is of the form $Bel_a^{\leq r}(\psi)$ (or another modal operator). There, it computes a table T_ψ by calling GKMOVE recursively on ψ, creates a copy of the Bayesian Network fragment defined by $G_a, \mathcal{P}_a$ with variables X^a, X^h, updates this network with values available in T_ψ, and performs a VE on that Bayesian Network. The set of variables returned there is Γ_ψ.

The VE performed on this BN fragment first identifies a set of variable symbols Γ_φ such that their corresponding variables in X^a, Γ_φ^a, will separate future computations from the one for ψ here. It then performs standard VE (e.g. [Dechter, 1996]) on the Bayesian Network such that Γ_φ^a remains.

The result of this VE is a table that includes probabilities for ψ conditioned on different values of Γ_φ^a. We compare those with r and build the resulting table T_φ which we return together with Γ_φ (the random-variable symbols in Φ which correspond to Γ_φ^a, i.e. the same symbols only without superscript a).

The computational benefit of this method (which we examine below) comes in two forms. First, let X_{rest}^a be those variables in X^a that are not in Γ_φ^a. Let X_{rest}^h be the descendants of X_{rest}^a that are not in Γ_ψ. Then summing out X_{rest}^h and X_{rest}^a results in a constant 1, regardless of the distribution governing X_{rest}^a (which remains unspecified in this BN fragment). So, we can easily remove them from the BN fragment. Second, the VE in the BN can be done efficiently with traditional methods if the BN has a bounded, small treewidth [Amir, 2010].

4.1.1 Algorithm GKMOVE Detailed

Subroutine GKMOVE works by recursion as follows.

If φ is an atomic formula then, then GKMOVE returns structures $\langle \Gamma, T \rangle$ with $\Gamma = \emptyset$ (when $\varphi = \top$ or $\varphi = \bot$) or Γ with the single random-variable symbol that appears in φ.

When φ is of the form $\neg\psi$, GKMOVE calls recursively for GKMOVE with ψ to compute the structure $\langle \Gamma_\psi, T_\psi \rangle$. Then, it performs the logical negation operation on T_ψ, which returns a table that has the rows of T_ψ updated with opposite truth values in the last column. It returns the resulting table together with the same set of variables.

When φ is $\psi \vee \xi$, the algorithm recursively computes tables for ψ and ξ. If combines these two truth table as follows. Let truth table for ψ be $\langle \Gamma_1, T_1 \rangle$ and truth table for ξ be $\langle \Gamma_2, T_2 \rangle$. GKMOVE returns a truth table whose variables is $\Gamma_1 \cup \Gamma_2$ and its values is disjunction of the values of the two truth table. Similarly we can compute the truth table for formula $\neg\psi$.

Finally, when φ is $Bel_a^{\leq r}(\psi)$, GKMOVE recursively computes the table T_ψ and set of variable symbols $\langle \Gamma, T \rangle$ for ψ. Let X^a and X^h be the random variables denoting the *actual state* and *possible state* (notions termed in Definition 10) of agent a respectively. Also assume that Γ^h is the set of variable symbols in X^h that corresponds to variables in Γ. Now, we define Γ_φ^a as the set of variables in X^a that are ancestors of variables in Γ_ψ.

Then, for every value vector γ_φ^a for Γ_φ^a we compute $pr(\gamma_\varphi^a) := (Pr(\psi^h \mid \Gamma_\varphi^a = \gamma_\varphi^a)$, where ψ^a is the formula ψ whose variable symbols are replaced with respective variable symbols with superscript h. We compare $pr(.)$ to r and create the table T_φ. Finally, GKMOVE returns $\langle \Gamma_\varphi, T_\varphi \rangle$.

Theorem 20 (Soundness) Let φ be a formula in PBBL, M a GKM, and s a state. The row of the table that GKMOVE returns that corresponds to s has value true iff $(M, s) \models \varphi$.

PROOF We prove this by structural induction on φ. We assume that the theorem holds for all sub formulas of φ. GKMOVE returns true and false for $\top$ and $\bot$ respectively which is correct. If φ is an atomic formula $X = v$, GKMOVE returns $\langle\{X\}, T\rangle$ and computes the truth value of $X = v$ for all different values of X which is correct.

If φ is $\psi \vee \xi$ then the algorithm recursively computes the truth table for ψ and ξ ($\langle\Gamma_\xi, T_\xi\rangle$ and $\langle\Gamma_\psi, T_\psi\rangle$). GKMOVE returns a truth table whose variables are $\Gamma_\psi \cup \Gamma_\xi$ and its values are disjunction of the values of the values appearing in the two truth table. For any $s \in S$ the values appearing in the table are the disjunction of corresponding values of ψ and ξ. A similar argument holds for $\varphi = \neg\psi$.

If φ is equal to $Bel_a^{\leq r}(\psi)$, GKMOVE recursively computes the truth table for ψ ($\langle\Gamma_\psi, T\rangle$) by calling Subroutine BelVE. By Definitions 5 and 11 we know that $(M, s) \models \varphi$ holds if $\sum_{s' \in S} \mathcal{P}_a(X^h = s' | X^a = s)\psi(s') \leq r$.

At Steps 2-3 of Subroutine BelVE we create a BN B using G_a and $\mathbf{C}_a$. Now we add a node X_ψ to this BN. The parents of ψ are Γ_ψ^h. Note that Γ_ψ^h is the set of variable symbols in X^h that corresponds to variables in Γ_ψ. We set the CPD on X_ψ so that it agrees with T. In this BN, $Pr(X_\psi = true | X^a = s) = \sum_{s' \in S, \psi(s') = true} \mathcal{P}_a(X^h = s' | X^a = s)$.

In B, X_ψ is d-separated from $X^a \setminus \Gamma_\varphi^a$. Let $X_1, ..., X_u$ be the variable symbols in Γ_φ^a. Then,

$$Pr(X_\psi = true | X^a = s) = Pr(X_\psi = true | \Gamma_\varphi^a = \langle s_1, ..., s_u\rangle)$$

for every $s \in S$ where we choose $s_1 = X_1(s), ..., s_u = X_u(s)$.

So, $(M, s) \models Bel_a^{\leq r}(\psi)$ if $\sum_{s' \in S} \mathcal{P}_a(X^h = s' | X^a = s)\psi(s') \leq r$ which is equal to $Pr(X_\psi = true | X^a = s) \leq r$. Thus returning truth value for all possible value vectors γ_φ^a of Γ_φ^a using Steps 6(a),6(b) is correct. ■

There are several simple ways to speed up this function. For example, there are methods to encode the tables in this program in much smaller tables. For example, when $\varphi = (X = v)$ and $|dom(X)| = n$, we can hold only two rows, one for value v and another one for $\neg v$. One should keep in mind though that this may complicate merging two truth tables.

Proposition 21 Let q be a query, s be a state, v be the maximum size of the domain of random variables. Function GKMOVE calculates the value of q on s in $O(v^t)$ time, for t the largest number of columns in a table generated through the algorithm.

The worst-case running time of this algorithm is the same as the running time of GBU of Section 3.1.

4.2 GKM Sampling

In this section we provide a sampling method for answering queries on GKMs.

Function ApxGKMOVE shown in Figure 4.4 presents our sampling method. We use a similar method to GKMOVE but instead of using a form of dynamic programming for computing an entire table of values at each stage, we sample the states. For queries with no modal operator function, ApxGKMOVE calculates the value of φ on s precisely (using the ToDo algorithm which is embedded here) and returns 1 if $\varphi = true$ and 0 otherwise.

For queries with modal operators such as $\varphi = Bel_a^{\leq r}(\psi)$, ApxGKMOVE repeats the following step n times for each modal operator (thus compounding the number of samples exponentially when modal operators are nested in φ). It samples values for X^h. For each sampled value vector (state) ApxGKMOVE recursively computes the value of ψ on this state.

Theorem 22 (Soundness of ApxGKMOVE) For GKM M, state s in M, and formula φ in PBBL, ApxGKMOVE(M,s,φ) returns a value in $\{0, 1\}$. Assume further that all modal operators in φ are of the form $Bel^{\leq r}$, and that for every subformula $\xi = Bel^{\leq r}(\psi)$ of φ, $M, s' \not\models Bel^{=r}(\psi)$ (NOTICE: "=" intended here).

Then, if $M, s \models \varphi$, its returned value converges to 1 in probability as $n \to \infty$. Similarly, If $M, s \not\models \varphi$, its returned value converges to 0 in probability as $n \to \infty$.

PROOF We apply the result of Theorem 17. We show that the probability distribution that we sample from and the number of samples and the calculation is the same as ApxToDo and therefore when the number of samples goes to infinity the returned value converges to the correct value.

The proof is with structural induction on φ. If φ is $\top$, $\bot$, or $X = v$ we return the actual value. Therefore the theorem holds. If φ is $\neg\psi$ both algorithms compute the value of ψ and return the negation of that. If φ is $\psi \vee \xi$ both algorithms compute the values of ψ and ξ and return 1 if one of the values is 1.

If φ is $Bel^{\leq r}(\psi)$ both sample from a uniform distribution over S and compute the value of ψ on those states. They both then use those values in the same way to compute the result. The only difference here is that the computation of value of $\mathcal{P}_a(X^h = s_i \, | X^a = s)$ is done on the BN instead of explicitly creating the subjective probabilities for all s, s'. ■

Proposition 23 Let φ be a query whose truth value on state s in GKM M we want to compute. Let m be the nesting height of modal operators in φ, and let n be the number of sampled states. Function ApxGKMOVE terminates in time $O(|\Phi| \cdot |\varphi| \cdot n^m)$.

PROOF The proof follows in an almost identical manner to the proof of Proposition 16. The main difference is the time taken to evaluate $\mathcal{P}_a(X^h = s_i \mid X^a = s)$, which now takes time $O(|\Phi|)$, the number of conditional probability terms in our product of factors. Notice that we can do this in linear time because X^a are all the parents of X^h, and there are no other variables) ■

Function ApxGKMOVE takes more time than Function ApxToDo because of the need for computation of conditional probability $\mathcal{P}_a(X^h = s_i \mid X^a = s)$. We pay this price because we save on space, which is exponentially larger when using ApxToDo. Note that this is due to the size of the tables $P_{a,s}(s')$.

4.3 Example

Consider the query $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ in our Holdem example as described in the previous chapters where players Alice and Bob are represented by a and b respectively. Again, this query refers to "according to Bob, with high probability (at least 0.9) according to Alice the probability of Bob having Ace of hearts is low (at most 0.2)".

Assume that we want to compute the above satisfaction query for the GKM defined in 2.4 and on state s that corresponds to $A\diamondsuit K\diamondsuit\ K\spadesuit 3\clubsuit\ K\clubsuit K\heartsuit Q\heartsuit 3\spadesuit 2\clubsuit$. This query is calculated as follows:

$$\Big(\sum_{s'\in S} \mathcal{P}_b(X^{h_1} = s' \mid X^{a_1} = s)\Big(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)(s')\Big)\Big) \geq 0.9$$

which is:

$$\Big(\sum_{s'\in S} \mathcal{P}_b(X^{h_1} = s'' \mid X^{a_1} = s) \\ \Big(\big(\sum_{s''\in S}(X_3(s'') = A\heartsuit \vee X_4(s'') = A\heartsuit) \\ \mathcal{P}_a(X^{h_2} = s'' \mid X^{a_2} = s')\big) \leq 0.2\Big)\Big) \geq 0.9$$

Notice that we created a fresh random variable for each Bel in the query.

To compute this satisfaction query we use GKMOVE. In the above formula we cannot move $\sum_{s'\in S}$ inside $\sum_{s''\in S}$, since the latter participates in an inequality. GKMOVE performs variable elimination on this formula in two rounds. It eliminates variables s'' in the first round and variables s' in the second round.

GKMOVE is a recursive function which first computes a table for $X_3 = A\heartsuit$ and a table for $X_4 = A\heartsuit$ as follows:

X_3	$X_3 = A\heartsuit$
$A\heartsuit$	$true$
$2\heartsuit$	$false$
.	.
.	.
.	.
$K\diamondsuit$	$false$

X_4	$X_4 = A\heartsuit$
$A\heartsuit$	$true$
$2\heartsuit$	$false$
.	.
.	.
.	.
$K\diamondsuit$	$false$

Then it computes a table for $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ as follows.

X_3	X_4	$X_3 = A\heartsuit \vee X_4 = A\heartsuit$
$A\heartsuit$	$A\heartsuit$	$true$
$A\heartsuit$	$2\heartsuit$	$true$
.	.	.
.	.	.
.	.	.
$A\spadesuit$	$A\spadesuit$	$false$
$A\spadesuit$	$2\spadesuit$	$false$
.	.	.
.	.	.
.	.	.
$K\diamondsuit$	$Q\diamondsuit$	$false$
$K\diamondsuit$	$K\diamondsuit$	$false$

Each row shows the value of the column variable and the final column is the value of sub-formula for those value combinations. The value of a sub-formula on state s only depends on the value of these variables.

Now to compute $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ we construct the bottom BN fragment of Figure 4.5. As you can see since the value of $X_3 = A\heartsuit \vee X_4 = A\heartsuit$ depends only on X_3 and X_4, we add a node whose parents in the possible state are X_3^h and X_4^h. Note that this BN corresponds to Alice. The ancestors of this new node are $X_1^a, X_2^a, X_5^a, \ldots, X_9^a$. Therefore for each possible combination of values for these variables we compute $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ using variable elimination and return a table whose variables are $X_1^a, X_2^a, X_5^a, \ldots, X_9^a$.

In the same manner we compute the value of $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ using BN fragment shown in the top part of Figure 4.5.

Using GKMOVE we answer queries with higher height of modal operators such as $Bel_a^{=1}(Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)))$. The BN fragments for this query is shown in Figure 4.6, 4.7 and 4.8. The BN node creation is as follows. To compute $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$ we copy the BN fragment of Alice (as shown in Figure 4.8) and add a node whose parents are X_3 and X_4. Since the ancestors of this node are X_1, X_2, $X_5, \ldots, X_9$, the parents of the new node to compute $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ are those nodes (as shown in Figure 4.7). Notice that this is added to BN fragment for Bob.

At the end, to compute $Bel_a^{=1}(Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)))$ we add a new node for $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$ to BN fragment of Alice as shown in Figure 4.6. The parents of this node in the BN fragment of Alice are the ancestor of the new node of BN fragment of Bob (shown in Figure 4.7).

4.4 Experimental Results

In this section we compare the running time of all our algorithms on our two-player Poker example. The queries that our algorithms evaluate are the followings:

$$
\begin{aligned}
&X_3 = A\heartsuit \vee X_4 = A\heartsuit \\
&Bel_a^{\leq r_1}(X_3 = A\heartsuit \vee X_4 = A\heartsuit) \\
&Bel_b^{\leq r_2}(Bel_a^{\leq r_1}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)) \\
&Bel_a^{\leq r_3}(Bel_b^{\leq r_2}(Bel_a^{\leq r_1}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))) \\
&\ldots \\
&Bel_b^{\leq r_10}(\ldots Bel_b^{\leq r_2}(Bel_a^{\leq r_1}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)))
\end{aligned}
$$

We run ToDo, KBU, and ApxToDo on a Poker PKM for queries with different heights of nested modal operators as explained above. As shown in Figure 4.9, the running time of the ToDo algorithm grows exponentially with the number of nested modal operators while KBU grows linearly. For this example the number of states in the state space is 47^4 so computation with many queries is practical in all methods, and it takes less than a second to compute a formula with no modal operator on all states. Therefore, KBU is a better option for this application. However, if we know that the height of queries of interest are limited and the state space is large, ToDo might be a faster option.

We also experiment with ApxToDo with different number of samples. The number next to ApxToDo is the number of samples. We sample the states uniformly from the state space. As shown in the figure, even with 100 samples, ApxToDo is tractable for queries of height at most 4. Usually in an application like Poker, the height of queries of interest is not more than 4. ApxToDo is also exponential in height of nested modal operators. Note that in typical real-world situations the degree of nesting in queries is small (*e.g.,* in Poker a player at most cares about what the opponent knows about what the player knows).

We also compare the running time of GKMOVE and ApxGKMOVE with KBU, ToDo, and ApxToDo. This confirms the theoretical results of section 3, 4 about the running time of our algorithms. As shown in the figure, GKMOVE and KBU grow linearly with the height of nesting and they both take less than a second to compute our satisfaction queries of size up to 10. Both approximate methods (ApxToDo and ApxGKMOVE) grow exponentially with the height of nesting. The running time of these methods are similar for our example, so we only show one in the figure. The difference between ApxGKMOVE and ApxToDo is that ApxGKMOVE saves in space while keeping computation time at a similar comparable $O(|\Phi| \cdot l \cdot n^m)$ compared to $O(l \cdot n^m)$ for $|\Phi|$ being the number of variable symbols defining the domain. Here $|\Phi|$ is 9.

```
**Subroutine** GKMOVE(PKM $M$, Query $\varphi$)

  1. if $\varphi = \top$ return $\langle \emptyset, true \rangle$; else if $\varphi = \bot$ return $\langle \emptyset, false \rangle$
  2. if $\varphi$ is an atomic formula $X = v$ then
       • return $\langle \{X\}, T \rangle$ where $T$ is a table with two columns labeled $X$
         and $f$ respectively, with rows $\langle v, true \rangle$ and $\langle v', false \rangle$ for every $v' \in$
         $dom(X)$ such that $v' \neq v$.
  3. if $\varphi = \neg\psi$ then
       (a) Set $\langle \Gamma_\psi, T_\psi \rangle \leftarrow$ GKMOVE($M, \psi$)
       (b) return $\langle \Gamma_\psi, \text{NEGATION}(T_\psi) \rangle$
  4. if $\varphi = \psi \vee \xi$ then
       (a) Set $\langle \Gamma_\psi, T_\psi \rangle \leftarrow$ GKMOVE($M, \psi$)
       (b) Set $\langle \Gamma_\xi, T_\xi \rangle \leftarrow$ GKMOVE($M, \xi$)
       (c) return $\langle \Gamma_\psi \cup \Gamma_\xi, \text{DISJUNCTION}(T_\psi, T_\xi) \rangle$
  5. return BelVE($M$,$\varphi$)
```

```
**Function** sGKMOVE(PKM $M$, State $s$, Query $\varphi$)

  1. Set $\langle \Gamma_\varphi, T_\varphi \rangle \leftarrow$ GKMOVE($M, \varphi$)
  2. $\Gamma_\varphi = \{X_1, ..., X_u\}$ for some $X_1, ..., X_u \in \Phi$.
  3. Find $\langle x_1, ..., x_u, t \rangle \in T$ such that $X_i(s) = x_i$ for all $i \leq u$
  4. return $t$
```

Figure 4.1: Ordered Variable Elimination for GKM (GKMOVE) algorithm. This algorithm receives M, φ and returns a table that maps value vectors for a restricted set of variables to $\{true, false\}$. sGKMOVE evaluates $M, s \models \varphi$ by searching in the table returned from GKMOVE for the variable values corresponding to s.

```
Subroutine NEGATION(Table $T$)
    1. Set $T_{neg} \leftarrow \emptyset$
    2. for every row $\langle x_1, ..., x_u, true \rangle$ in $T$ for some $x_1, ..., x_u$, add a row $\langle x_1, ..., x_u, false \rangle$ to $T_{neg}$.
    3. for every row $\langle x_1, ..., x_u, false \rangle$ in $T$ for some $x_1, ..., x_u$, add a row $\langle x_1, ..., x_u, true \rangle$ to $T_{neg}$.
    4. return $T_{neg}$
```

```
Subroutine DISJUNCTION(Table $T_1$, Table $T_2$)
    1. Set $T_{disj} \leftarrow T_1 \bowtie_{\text{all columns but } f} T_2$, where $\bowtie_{\text{all columns but } f}$ is the natural join on database tables that keeps both columns from $T_1$ and $T_2$ labeled $f$ (we name them $f_1, f_2$ in $T$).
    2. Create column $f$ in $T_{disj}$, and for every row $\langle x_1, ..., x_u, t_1, t_2 \rangle \in T_{disj}$ set the $f$-column value to the truth value of $t_1 \vee t_2$
    3. Drop columns $f_1, f_2$ from $T_{disj}$
    4. return $T_{disj}$
```

Figure 4.2: Subroutines NEGATION and DISJUNCTION perform logical negation and disjunction on the f columns of their respective tables.

```
**Subroutine** BelVE(GKM $M$, Query $\varphi$)

1. $\varphi$ is of the form $Bel_a^{\leq r}(\psi)$; Set $T_\varphi \leftarrow \emptyset$; Set $\langle \Gamma_\psi, T_\psi \rangle \leftarrow$ GKMOVE($M, \psi$)
2. Let $B$ be a copy of the BN fragment over $X^a$ and $X^h$ for Agent $a$ in $M$
3. Change $B$ as follows: create a new node $X^h_\psi$ with parents $\Gamma^h_\psi$, where $\Gamma^h_\psi$ is
   $\Gamma_\psi$ with superscript $h$ added to the variable symbols (thus, now designating
   variables in $B$); for every vector of values $\gamma^h_\psi$ to $\Gamma^h_\psi$, if $T_\psi(\Gamma^h_\psi = \gamma^h_\psi) =$
   $false$, set the CPD value for $X^h_\psi$ to $0$. For all other values $\gamma^h_\psi$ to $\Gamma^h_\psi$ set the
   CPD value for $X^h_\psi$ to $1$.
4. Set $\Gamma^a_\varphi \leftarrow$ ancestors($\Gamma^h_\psi$) in $B$
5. Set $\Gamma_\varphi$ the set of variable symbols in $\Phi$ that correspond to $\Gamma^a_\varphi$ (i.e. without
   superscript $a$)
6. **for** every vector of values $\gamma^a_\varphi$ to $\Gamma^a_\varphi$ **do**
    (a) $p \leftarrow VE(B|\Gamma^a_\varphi = \gamma^a_\varphi)$
    (b) Let $t \leftarrow true$ if $p \leq r$, and *false* otherwise.
    (c) Add $\langle \gamma^a_\varphi, t \rangle$ to $T_\varphi$
7. **return** $\langle \Gamma_\varphi, T_\varphi \rangle$
```

Figure 4.3: Subroutines NEGATION and DISJUNCTION perform logical negation and disjunction on the f columns of their respective tables.

```
FUNCTION ApxGKMOVE(GKM $M$, State $s$, Query $\varphi$)
  1. if $\varphi = \top$ return $1$; otherwise if $\varphi = \bot$ return $0$
  2. if $\varphi$ is an atomic formula $X = v$ then
       • return 1 if $X(s) = v$ and 0 otherwise
  3. if $\varphi = \neg\psi$ then
       • return $1-$ ApxGKMOVE($M, s, \psi$)
  4. if $\varphi = \psi \vee \xi$ then
       (a) $p_\psi \leftarrow$ ApxGKMOVE($M, s, \psi$)
       (b) $p_\xi \leftarrow$ ApxGKMOVE($M, s, \xi$)
       (c) if return $1 - ((1 - p_\psi) \cdot (1 - p_\xi))$
  5. It must be that $\varphi = Bel_a^{\leq r}(\psi)$; Set $T \leftarrow$ empty list (possibly with repetitions or contradicting entries);
  6. for $j \leftarrow 1$ to $n$
       (a) Select $s_j \in S$ an independently drawn sample from a uniform distribution over $S$
       (b) Set $p_\psi \leftarrow$ ApxGKMOVE($M, s_j, \psi$)
       (c) Add $\langle s_j, p_\psi \rangle$ to $T$ at position $j$ (notice that we allow $T$ to have duplicate entries or contradicting entries (ones in which the same state receives different truth values).)
  7. Let $B$ be a copy of the BN fragment over $X^a$ and $X^h$ for Agent $a$ in $M$
  8. Set $p$ to 1 if $\frac{|S|}{n} \times \sum_{i=1}^{n} T(i)\mathcal{P}_a(X^h = s_i \,|X^a = s) \leq r$ holds and 0 otherwise, where we compute $\mathcal{P}_a(X^h = s_i \,|X^a = s)$ by the definition (product of conditional probabilities) of $\mathcal{P}_a$ in Definition 11.
  9. return $p$
```

Figure 4.4: ApxGKMOVE sampling algorithm.

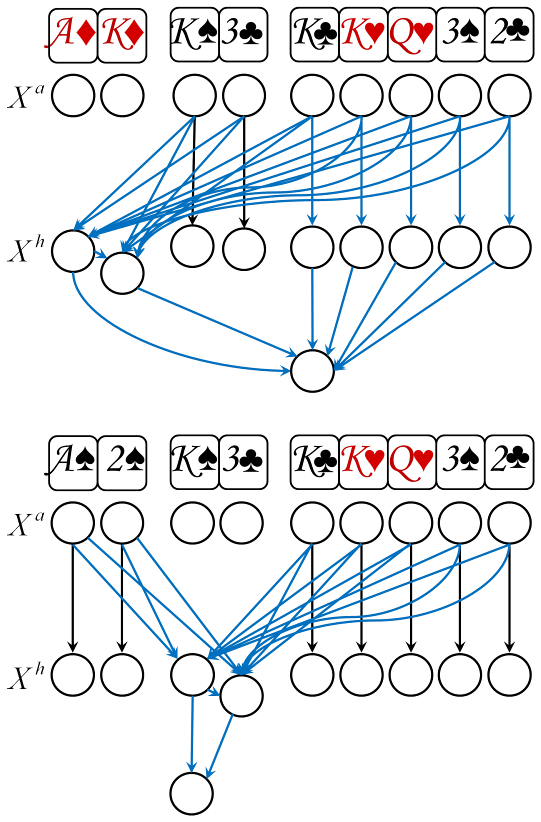

Figure 4.5: BN fragments that GKMOVE creates for computing the truth value of $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$. The bottom node in the upper fragment is $X^h_{Bel_a^{\leq 0.2}(X_3=A\heartsuit \vee X_4=A\heartsuit)}$. The bottom node in the lower fragment is $X^h_{X_3=A\heartsuit \vee X_4=A\heartsuit}$. The function first computes the lower fragment and then the upper fragment.

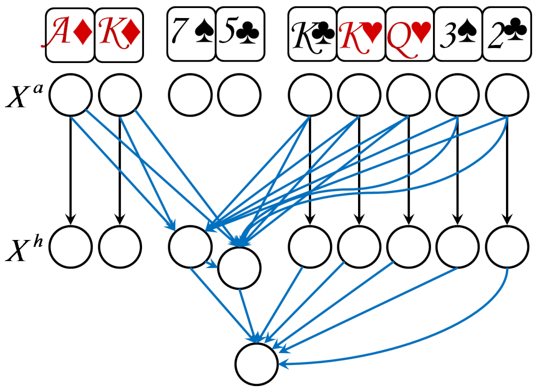

Figure 4.6: Alice's BN fragment for $Bel_a^{=1}(Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)))$. The bottom node $X^h_{Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3=A\heartsuit \vee X_4=A\heartsuit))}$.

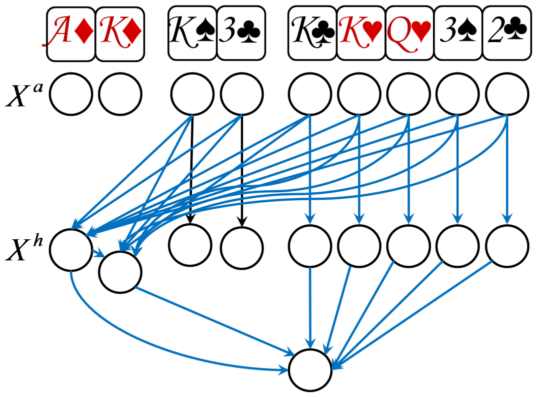

Figure 4.7: Bob's BN fragment for $Bel_b^{\geq 0.9}(Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit))$. The bottom node $X^h_{Bel_a^{\leq 0.2}(X_3=A\heartsuit \vee X_4=A\heartsuit)}$.

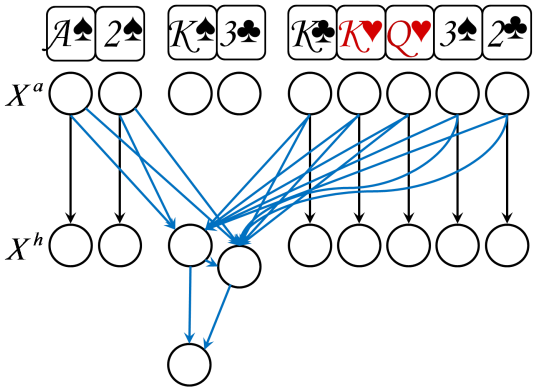

Figure 4.8: Alice's BN fragment for $Bel_a^{\leq 0.2}(X_3 = A\heartsuit \vee X_4 = A\heartsuit)$. The bottom node $X^h_{X_3=A\heartsuit \vee X_4=A\heartsuit}$.

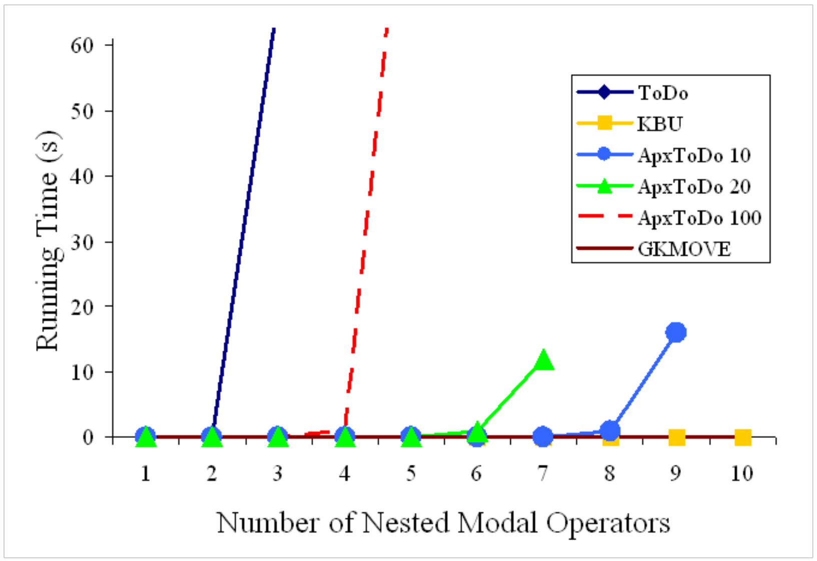

Figure 4.9: Running time.

CHAPTER 5

RELATED WORK

Our work is most closely related to works on combining modal logics with notions of probability or counting. Further, it takes ideas and is relevant to works on combining FOL and probabilities, nested probabilistic knowledge in dynamic domains, modeling languages for probabilistic programs, and decisions in POMDPs.

5.1 Combining Modal Logics with Probabilities

5.1.1 Reasoning with one Model

Our work is most closely related to the work of Milch and Koller [Milch and Koller, 2000] (MK hereforth). Compared to MK, the easiest distinction might be that our work is the first to consider sampling for inference in such models. This is not considered by MK's framework, or by the frameworks of others working on combining modal logic and probability. Comparison with respect to other factors of this thesis is detailed below.

As we pointed out in Chapter 2, our work is similar to theirs in that we use a similar language. Their work and ours derive semantics as a restricted form of semantics presented by [Fagin and Halpern, 1994] and that can be traced back to [Harsanyi, 1967] (in the context of *type spaces* and multi-agent games rather than modal logics). Our language is more restricted than MK's in that we do not include observations as an integral part.

The semantics of MK applies *observation sets* that are used to derive an agent's subjective probability from a prior on world states shared by all agents. They connect those observation sets later to models for multi-agent decisions based on influence diagrams, MAIDs (a concept introduced more generally in later papers). In MAIDs observation sets refer to observations assumed to be available to agents in influence diagram.

While not incorporating observations into our semantics, our semantics is more general than that of MK in that it allows agents that have different subjective probability models. The agents' models do not even need to be derived by conditioning an agent-specific prior. As pointed out in Section 2.3, there are ways for each system (MK's and ours) to simulate the other one, but those are limited and have computational ramifications.

With regards to computational considerations, there again is a close relationship between the two works. MK's reasoning is done in the model as is ours. As we do, they also describe algorithms for inference with a Bayesian Network (BN) representation of joint probability available in their models. However, there are important differences between their models' assumptions and ours, and hence the computational properties that result are different.

Specifically, MK use BNs to represent the prior shared by all agents about the world states. In contrast, our work uses BNs to represent the subjective probabilities of agents, namely, the probability held by each agent that he is at s' when in fact he is in s. If we take a model described in MK's framework, and wish to convert it into a GKM, we need to perform inference in MK's BN, and then construct a different BN to represent the resulting subjective distribution (which may have very different independence properties than the prior).

The other direction (from ours to theirs) is not directly possibly (though may be possible by adding variables to their model). Thus, sans observations, our algorithms can address models in their framework.

Reasoning within a model relies on semantics first presented by [Harsanyi, 1967] in the context of *type spaces* and multi-agent games rather than modal logics. The main difference between our work and that literature is that these models are extended and used to examine different questions in game theory in an ad-hoc fashion. The examination of those models in the framework of a logic (hence, permitting automated reasoning about them) has been developed in the last 15 years [Heifetz and Mongin, 1998; Aumann, 1999a; 1999b; Heifetz and Mongin, 2001; Aumann and Heifetz, 2002], but has taken an approach that seeks to find properties of such models in general and reason about them. We examine this line of work in more detail in Section 5.1.2.

5.1.2 Axioms and Validity

A body of works [Fattorosi-Barnaba and Amati, 1987; Fagin and Halpern, 1994; Heifetz and Mongin, 1998; 2001; Cao, 2007; Ferreira *et al.*, 2008] is concerned more broadly than above on creating formal systems (including language, models, axioms, and decision procedures where possible) that represent the knowledge of agents about other agents' knowledge appropriately. These represent uncertainty and shortage of knowledge as a probability.

An expressive logic presented by Fagin and Halpern in [Fagin and Halpern, 1994] extends work by [Fagin *et al.*, 1990] and includes both an accessibility relation and probability measures, and is able to represent and discuss in the logic probabilities of statements and also linear combinations of those. Fagin and Halpern present a sound and complete axiomatization and several extensions that capture conditions of interest to works in distributed systems, cryptography, and analysis of probabilistic programs.

They show that their system has the finite-model property (if there is a model satisfying some axioms, there is also a finite-sized model satisfying the same axioms), hence reasoning there is decidable. Further, they show that finding if φ, a formula in their language, is valid is EXPTIME-complete in their system with or without most of their extended sets of axioms. For two subsets they show better tractability, namely, PSPACE completeness and NP completeness (when reasoning about a single agent), respectively. [Fagin *et al.*, 1990] shows that satisfiability for a more restricted language without the combination of nested modalities and linear combinations of probabilities (that still includes ours) is NP-complete.

In contrast to these works, our work is limited to reasoning with a given model, so our task is much simpler. This is most evident by observing the relatively low (typically linear in $|\varphi|$ instead of $exp(|\varphi|)$ of [Fagin and Halpern, 1994]) computation time for our algorithms. Otherwise, our language and semantics can be seen as derived from theirs. One could consider using some of our techniques for reasoning with their logic, if representative models can be enumerated or sampled efficiently. This is beyond our scope here.

A related set of works rooted in economics [Heifetz and Mongin, 1998; 2001] seeks to formalize type systems [Harsanyi, 1967] and presents formal systems that can be seen as restrictions of those of [Fagin *et al.*, 1990; Fagin and Halpern, 1994]. Their language includes modal operators for comparing the probability of a formula with numbers in $[0, 1]$, in a manner very similar to the one used in this thesis. They provide an axiomatization that unlike [Fagin *et al.*, 1990; Fagin and Halpern, 1994] does not allow formulas involving linear combinations of probabilities. They examine several families of type systems, and show that their axiomatization for some such families is sound and complete, and their logic is decidable for those.

5.1.3 Other Combinations of Modal Logic with Probability-Like Structures

The strings of works on *graded modal logics* [Goble, 1970; Fine, 1972; Fattorosi-Barnaba and de Caro, 1985] includes a somewhat different modal operator that concerns the number of states accessible from a state (instead of a subjective probability distribution over states). These logics are the subject of research with applications in formal methods among others [Tobies, 2000; Kazakov and Pratt-Hartmann, 2009].

Clearly, the languages for these logics are different from ours and the body of works on probabilistic modal logics, but there are still similarities. In particular, using such logics one can represent a limited version of our semantics, namely, one in which we assume a uniform distribution as a subjective probability, and the size of the set provides the degree of certainty we attribute to each possible state.

In Figure 5.1, we summarize different representations for nested knowledge and nested probabilistic knowledge. In Figure 5.2, we show how different examples are represented using the syntax of different logics.

5.2 Combining First-Order Logic and Probabilities

Many applications of AI have both stochastic and non-stochastic elements. For example, robot control can include high-level specifications in logic and a lower-level probabilistic sensing model. Also, Natural Language Processing wishes to apply high-level knowledge in logic with lower-level probabilistic models of text and spoken signals. Many databases and database views are logical in nature, while relationships between those databases (e.g., if two database columns refer to the same concept) are uncertain.

The last 20 years have seen much work in the AI community and the Databases

Type	**Logic**	**Logic+Probability**
Syntax	$\Box$,$\Diamond$, propositional logic	$w_i , k_i \quad L_i(\phi) \geq \alpha$
Semantics	possible world semantics (states + accessibility relation)	probabilistic possible world semantics (states + probabilistic accessibility)
Interpretation	necessity, knowledge, obligation	probabilistic knowledge
Axiomatization/ Logic	e.g., K, D, T, S4, S5	AX(Fagin, Halpern) A+(Heifetx, Mongin)
Example Applications	what color hat am I wearing?	repeated rock-paper-scissors, agents in distributed systems
Real-World Applications	robot coordination, web agents reaching agreement	

Figure 5.1: Different Representations for Nested Knowledge.

community on the combination of first-order logic and probabilistic expressivity. These works present languages that can express probability distributions together with explicit references to objects, functions, and relationships, as in First-Order Logic (FOL) (e.g., [Nilsson, 1986; Halpern, 1990; Pfeffer *et al.*, 1999; Kersting and Raedt, 2000; Milch *et al.*, 2005]). These languages are useful frameworks for many machine learning applications, and recent works also show that they are useful for computational efficiency of inference [Poole, 2003; de Salvo Braz *et al.*, 2006].

Research on the combination of logic and probability is ongoing. Current challenges include (a) applying relational structure in speeding up inference and treating probabilistic models over many objects, (b) combining knowledge bases that are given already in probabilistic or logical form, and (c) extending representation languages to include functions and equality of objects in sound and simple ways.

The work reported in this thesis is similar to those works in that it includes

Examples	Modal Logic	Nested Probabilistic Knowledge FH	HM
Know that the coin is fair	$\square$ fair	$k(\text{fair})$ $k(w(\text{fair}) \geq 1)$	$L(\text{fair}) \geq 1$
Know that the probability of H is not less than ½	NA	$k(w(H) \geq \frac{1}{2})$	NA
Know with probability ½ that the coin is landed H	NA	NA unless we interpret k, w differently	$L(H) = \frac{1}{2}$

Figure 5.2: How different examples are represented using different syntaxes. Example: Assume that there is a coin that is either fair or two-headed. The coin is tossed and our agent does not see the outcome.

elements from logical representations and elements from probabilistic representations. It differs from those works in the elements of logic that are combined with probabilities and the probabilistic frameworks with which they are combined. We give some details in the following paragraphs.

A body of works [Pfeffer *et al.*, 1999; Kersting and Raedt, 2000; Jaeger, 1997; Ng and Subramanian, 1992; Ngo and Haddawy, 1995; Friedman *et al.*, 1999; Koller and Pfeffer, 1998] focuses on combining relational representations with Bayesian Network probability representations. These combinations take the form of logic programs extended with probabilistic interpretations of rules, or a frame-based view in which objects instantiate classes, and the objects may have uncertainty over elements or uncertainty over their connections. They all have a rather direct Bayesian Network interpretation. We (and our closer works mentioned above) differ from those works in that we do not consider relational aspects in logic but rather a Kripke-structure like accessibility relation which our models exchange for subjective probability distributions.

Another important body of works [Nilsson, 1986; Halpern, 1990; Bacchus *et al.*, 1994; Richardson and Domingos, 2006] focuses on combining FOL expressivity (quantified statements, set of models instead of a single model, functions) more directly with probability. Some of these works (e.g. [Nilsson, 1986; Halpern, 1990]) extend FOL with probability statements about FOL statements. [Halpern, 1990] noticed that there are two kinds of probabilistic statements that one may wish to have in such contexts, and he axiomatized the resulting logic, pointing out that some of them cannot be axiomatized in a recursively enumerable way. Others (e.g. [Bacchus *et al.*, 1994; Richardson and Domingos, 2006]) introduce assumptions such as maximum entropy or a known domain size on top of an FOL sentential representation and derive a single model. We differ from all of those works in that we do not consider first-order (not even relational) elements in our representation, as pointed out above.

5.3 Nested Probabilistic Knowledge in Dynamic Domains (When Actions Change the World State)

In games such as Poker and Rock-Paper-Scissors, each agent performs actions and receives observations during the course of the game that affect his belief state. Therefore, the model of the world needs to be dependent on the actions that are performed by agents. An example of such games is Kuhn's Poker - a simple game in which the players can benefit from reasoning about each other's probabilistic knowledge.

A line of research focuses on applying probabilistic knowledge to decision making [Milch and Koller, 2000; Koller, 2001; Koller and Milch, 2001; Milch and Koller, 2003]. These works on models called MAIDs model the dynamics of the system as an extension of Influence Diagrams by adding updates of agents' knowledge to the model and making agents' decisions take into account other agents' decisions (modeling the latter as decisions instead of random variables).

These works assume that (1) the agents do not forget observations they have made (*perfect recall assumption*) and (2) the value of the variables do not change from stage to stage. Another work in this category is [Gal and Pfeffer, 2008] on networks of influence diagrams (NIDs), which lift some of the common-prior and rationality assumptions of MAIDs.

Our work does not concern decisions and cannot represent observations as integral parts of our model. However, by compiling observations into our model, our work can serve as a subroutine in the representations of agents' knowledge and the inference needed about them throughout the solutions of MAIDs. We did not explore this direction further in this thesis, but we postulate that such an extension would be more expressive than present systems.

Different works provided models of opponents in dynamic games in which action and observation model are known or learned. [Richards and Amir, 2007; Bard and Bowling, 2007] use particle filtering as part of a modeling approach in which a dynamic opponent applies actions and changes the state of the world. [Bard and Bowling, 2007] assumes that the dynamic model of the system has a known structure (e.g. the distribution is Normal) and the parameters are unknown. [Richards and Amir, 2007] assumes that the transition model is fully known. Our work can fit within those by adding our more expressive agent models to those frameworks, yielding better opponent models.

A large group of works concerns modeling the knowledge of agents' in stochastic dynamic worlds. Closest to ours are works that use logic to represent and reason about such domains, e.g. [Bacchus *et al.*, 1999; Kooi, 2003; van Benthem *et al.*, 2009; Sack, 2009]. These works consider updating the probabilistic knowledge of agents after performing actions, where *probabilistic knowledge* means a subjective distribution over possible world states as considered by [Fagin *et al.*, 1990].

[Bacchus *et al.*, 1999] presented a simple axiomatization that captures an agent's state of belief and the manner in which these beliefs change with (deterministic or probabilistic) actions. They build on a general logical theory of action developed by Reiter and others [Reiter, 2001], formalized in the situation calculus [McCarthy and Hayes, 1969]. They add a few fluents including K to situation calculus in which $K(s, s')$ means situation s and s' are indistinguishable to the agent (as in modal logic). Then they quantify the notion of possibility (K fluent) by associating with each world state the agent considers possible the agent's degree of belief that that is the actual world.

Our work differs from these in that we have a single (static) probabilistic model, whereas their framework is that of situation calculus, and reasoning is about a set of models. For technical reasons their model of agents' knowledge in a situation is not required to be a probability, but there is a simple normalization that brings it into the same frameworks as discussed above. Our work can be combined with this line of works and help make them more computationally tractable by assuming a single a single probabilistic model for actions and knowledge, and reasoning in that model.

The works of [Kooi, 2003; van Benthem *et al.*, 2009; Sack, 2009] take a similar approach to [Bacchus *et al.*, 1999], but are focused further on probabilistic beliefs over beliefs in *dynamic logic*. These works apply semantics that is more general than ours in that it includes both a subjective probability per agent per state and also an accessibility relation. The language of [van Benthem *et al.*, 2009] includes both a knowledge (all accessible worlds) and a probabilistic (probability equals to p) modal operator, but not a comparison operator (probability at most p). They consider actions that change agents' knowledge by observation models, and they present complete axiomatizations of these languages, but do not consider complete computational issues with reasoning in a model. Our results may apply to their system, if our semantics is extended properly.

5.4 Languages that Model Probabilistic Programs

Propositional probabilistic variants of temporal logic were studied in [Lehmann and Shelah, 1983; Hart and Sharir, 1984]. These works' semantics is similar to the one we adopt here, but the focus on temporal verification of programs yields a different language. Specifically, they are interested in formulas about infinite paths of states (e.g., paths along which either proposition q always holds or else q holds until the first time r holds). These models are mainly used to analyze transition from state to state and a path in such models is interpreted as temporal transition between states. The main application that is discussed in these papers is analyzing probabilistic programs.

A more expressive semantics and syntax are introduced by [Kozen, 1983; Feldman, 1986] for a similar purpose, namely, the verification and analysis of probabilistic programs. Their syntax is richer than [Lehmann and Shelah, 1983; Hart and Sharir, 1984] in that they allow explicit probability statements rather than just talking about probability 0 and 1. However, they do not consider beliefs over beliefs in the way that we (and closer works to ours) do. Therefore, the two languages are not easily comparable.

In these systems transition models are probabilistic. Following those works, the probabilistic-programs verification community focused on nondeterminism (*demonic uncertainty*) instead of probabilistic. Recent works (e.g. [Morgan and McIver, 1999]) revive the old topic by combining probabilistic and nondeterministic uncertainty, but the probabilistic parts of those models and methods remain the same.

5.5 Related Works on Partially Observable Markov Decision Processes(POMDPs)

A group of related work models agents' beliefs about other agents' beliefs to a finite level of nesting. This includes work on interactive POMDPs, such as [Gmytrasiewicz and Doshi, 2005; Doshi *et al.*, 2009]. They extend the framework of partially observable Markov decision processes (POMDPs) to multi-agent settings by incorporating the notion of agent models into the state space. Agents maintain beliefs over physical states of the environment and over models of other agents, and they use Bayesian update to maintain their beliefs over time. However, this only allows agents to reason about other agents' models and not deeper than that while our model does not restrict the height of nested modal operators.

Another line of work is work in the AI community on partially observable stochastic games and their cooperative counterparts, decentralized POMDPs, which includes [Hansen, 2004; Bernstein *et al.*, 2000; Zettlemoyer *et al.*, 2008]. Among these, only work of [Zettlemoyer *et al.*, 2008] focuses on filtering of infinitely nested beliefs. They define an infinite sequence of nested beliefs about the state of the world at the current time, and present a filtering algorithm that maintains a finite representation which can be used to generate these beliefs. The main difference between this work and ours is that we allow finite nested modal operators. On the other hand, in [Hansen, 2004; Bernstein *et al.*, 2000] policies are represented as direct mappings from observation histories to actions. That approach removes the need for the agents to perform any kind of filtering, but requires the specification of some particular class of policies that return actions for arbitrarily long histories.

CHAPTER 6

FUTURE WORK

In this chapter, we provide a few directions for future work. We provide motivating examples and background information for these directions.

One direction of future work is to implement a Poker player using this framework. This player can be played against other AI Poker players. Using algorithms in this thesis, the Poker player can use the knowledge he acquires in the course of the game to play better.

As shown in this thesis, a game such as Poker can be modeled with our framework. However we need to be able to model the actions of other players to be able to reason about a game. For example, a player should be able to change his model when he sees that the other player raises. In order to do that we need an action model that enables a player to transit from one model to another. Then in this new model we can use the same efficient algorithms developed here to answer satisfaction queries. However the transition model should be simple and should be efficient to reason with. Possibly filtering algorithms can be developed to transit from one model to another with an action.

Another direction of future work is defining a decision making model. A player can answer queries using our framework but he does not know how he can benefit from it or what should he do if the world model satisfies a query. A line of research exists that focuses on applying probabilistic knowledge to decision making [Milch and Koller, 2000; Koller, 2001; Koller and Milch, 2001; Milch and Koller, 2003]. They model the dynamics of the system as an extension of Influence Diagrams by making agents' decisions take into account other agents' decisions. Investigating this direction using our syntax and semantics is another area that can be explored in future.

Another direction is opponent modeling which can be viewed as learning formulas. This is useful when we know that a player makes his decisions based on the truth value of some formula. Learning that formula can help the first player to reason about the world. Assume that a player has some outside knowledge about the other player's strategy. For example assume that player 1 knows that player 2 raises when he believes that his probability of winning is greater than 0.7 and checks otherwise and when player 2 raises he folds if his probability of winning is less than 0.4 and calls otherwise. If we assume that player 1 knows that player 2 raises when he believes that his probability of winning is greater than some number and checks otherwise, learning that number is straightforward. However there are many other scenarios that might be the case in this game. For example, player 1 might raise if his winning probability is between two numbers A and B, and checks if his winning probability is greater than B to trap the other player into raising.

All of these possibilities can be modeled with a formula encoded in our language. However, learning such formulas is not straightforward. One direction of future work is to provide fast algorithms to learn such formulas. Then a player can use those to update his knowledge and use his knowledge to make decisions.

CHAPTER 7

CONCLUSION

In this thesis we provided a framework for representing and reasoning with probabilistic knowledge of agents about other agents. We provided a language that is based on [Fagin and Halpern, 1988] and we explained how it can be used for representing different examples. We also provided tractable exact and approximate reasoning algorithms for our model.

Our language can be used for representing nested probabilistic knowledge of agents about the world and about the knowledge of each other. The formulas represented with our language can have as many nested knowledge operators as we wish. Our reasoning algorithms can evaluate the value of a satisfaction query on a given state and a given model in a tractable time and space. If the number of states are large, our exact algorithms might be slow, therefore we provided approximate algorithms for such cases.

Our syntax and semantics for reasoning with nested probabilistic knowledge is an extension of previous models of probabilistic knowledge. We provide two different models, PKMs and GKMs. GKMs are a factored representation for probabilistic modal structures (PKMs). The main contribution is defining graphical Kripke models (GKMs). It provides a factored representation of a Kripke model without requiring the common prior assumption of Milch and Koller [Milch and Koller, 2000] (MK). In addition to removing this assumption, GKMs remove "observation assumption", which asserts that each agent has a prior distribution and an agent's probability distribution in each state can be derived by conditioning that prior on some observations. GKMs do not require an agent's probability distributions in different equivalence classes of states to be related to each other in any way.

We provided both exact and approximate algorithms for evaluating satisfaction queries on PKMs. We provided exact algorithms of two kinds. One of them is more efficient with larger state spaces, and the other one is more efficient with shallow nesting of modal operators in queries (i.e. the largest number of modal operators in root-to-leaf paths in expression trees of queries is small). We also introduced a sampling algorithm for PKMs, and showed that it converges to the correct answer under some conditions on the query formula. We proved that when this condition fails, the answers may not converge and sometimes are guaranteed (almost surely) not to converge.

We also introduced exact and approximate reasoning methods for answering queries on GKMs. Our exact method uses variable elimination to determine the value of the satisfaction query. Here, our model is more compact than PKMs and therefore enables larger-scale applications.

We discovered there are applications that modal logic is not good enough to deal with such as Poker. However, these applications can be treated with adding probability to modal logic. We also discovered that although theoretically it is straightforward to evaluate the value of a formula on a probabilistic knowledge model, experimentally we need to develop fast algorithms (exact/approximate) to scale up to large domains. It means that when we have very many states in our model, computing queries with nested modal functions is not tractable. Therefore to compute these queries we might need to sample the state space and estimate the value instead.

We also discovered that probabilistic independence assumptions can be ported to probabilistic modal logic models. In these models the accessibility relation can benefit from encoding with BN fragments. These independence assumptions can be used in developing tractable model checking algorithms.

We discovered that for the purpose of reasoning, it is easier to represent a world with a model than a set of axioms. To check if a query is implied with a set of axioms, we need to have a sound and complete axiomatization. However, if you represent a world with a model, the truth value of a query can be computed easily theoretically. We provided algorithms that can compute it fast as well.

In conclusion, we showed how the framework described in this thesis can be used to model a game such as Poker in which players reason about each others' probabilistic nested knowledge. Therefore, we can use this framework and reasoning algorithms to model and reason about games such as Poker in which players usually decide based on what they know about the state of the game and other players' knowledge.

REFERENCES

[Amir, 2010] Eyal Amir. Approximation algorithms for treewidth. *Algorithmica*, 56(1):448–479, 2010.

[Aumann and Heifetz, 2002] Robert J. Aumann and Aviad Heifetz. Incomplete information. In *Handbook of Game Theory with Economic Applications*, volume 3. Elsevier, 2002.

[Aumann, 1986] Robert J. Aumann. Reasoning about knowledge in economics. In *TARK '86: Proceedings of the 1986 conference on Theoretical aspects of reasoning about knowledge*, pages 251–251, San Francisco, CA, USA, 1986. Morgan Kaufmann Publishers Inc.

[Aumann, 1999a] Robert J. Aumann. Interactive epistemology i: Knowledge. *International Journal of Game Theory*, 28(3):263–300, 1999.

[Aumann, 1999b] Robert J. Aumann. Interactive epistemology ii: Probability. *International Journal of Game Theory*, 28(3):301–314, 1999.

[Bacchus *et al.*, 1994] Fahiem Bacchus, Adam J. Grove, Joseph Y. Halpern, and Daphne Koller. Generating new beliefs from old. In R. Lopez de Mantaras and D. Poole, editors, *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 37–45, San Francisco, CA, 1994. Morgan Kaufmann.

[Bacchus *et al.*, 1999] Fahiem Bacchus, Joseph Y. Halpern, and Hector J. Levesque. Reasoning about noisy sensors and effectors in the situation calculus. *Artificial Intelligence*, 111(1–2):171–208, 1999.

[Bard and Bowling, 2007] Nolan Bard and Michael H. Bowling. Particle filtering for dynamic agent modelling in simplified poker. In *Proc. National Conference on Artificial Intelligence (AAAI '07)*, pages 515–514, 2007.

[Bernstein *et al.*, 2000] Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of markov decision processes. In *Mathematics of Operations Research*, page 2002, 2000.

[Blackburn *et al.*, 2007] P. Blackburn, J. Van Benthem, and F. Wolter, editors. *Handbook of Modal Logic*. Elsevier, 2007.

[Blei *et al.*, 2003] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of machine Learning Research 3*, 3:993–1022, 2003.

[Cao, 2007] Zining Cao. Towards an epistemic logic for uncertain agents. In *Proceedings of the 5th international Central and Eastern European conference on Multi-Agent Systems and Applications V*, CEEMAS '07, pages 266–276, Berlin, Heidelberg, 2007. Springer-Verlag.

[de Salvo Braz *et al.*, 2006] Rodrigo de Salvo Braz, Eyal Amir, and Dan Roth. MPE and partial inversion in lifted probabilistic variable elimination. In *Proc. National Conference on Artificial Intelligence (AAAI '06)*. AAAI Press, 2006.

[Dechter, 1996] Rina Dechter. Bucket elimination: A unifying framework for probabilistic inference. In *Proc. Twelfth Conference on Uncertainty in Artificial Intelligence (UAI '96)*, pages 211–219. Morgan Kaufmann, 1996.

[Doshi *et al.*, 2009] Prashant Doshi, Yifeng Zeng, and Qiongyu Chen. Graphical models for interactive pomdps: representations and solutions. *Autonomous Agents and Multi-Agent Systems*, 18:376–416, June 2009.

[Fagin and Halpern, 1988] Ronald Fagin and Joseph Y. Halpern. Reasoning about knowledge and probability. In *TARK '88: Proceedings of the 2nd conference on Theoretical aspects of reasoning about knowledge*, pages 277–293, San Francisco, CA, USA, 1988. Morgan Kaufmann Publishers Inc.

[Fagin and Halpern, 1994] Ronald Fagin and Joseph Halpern. Reasoning about knowledge and probability. *Journal of the ACM*, 41(2):340–367, 1994.

[Fagin *et al.*, 1990] Ronald Fagin, Joseph Y. Halpern, and Nimrod Megiddo. A logic for reasoning about probabilities. *Information and Computation*, 87:78–128, 1990.

[Fagin *et al.*, 1995] R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning about knowledge*. MIT Press, 1995.

[Fattorosi-Barnaba and Amati, 1987] M. Fattorosi-Barnaba and G. Amati. Modal operators with probabilistic interpretations I. *Studia Logica*, 46:383–393, 1987.

[Fattorosi-Barnaba and de Caro, 1985] Maurizio Fattorosi-Barnaba and Francesco de Caro. Graded modalities I. *Studia Logica*, 44:197–221, 1985.

[Feldman, 1986] Yishai A Feldman. A decidable propositional dynamic logic with explicit probabilities. *Inf. Control*, 63:11–38, November 1986.

[Feller, 1968] William Feller. *An Introduction to Probability Theory and Its Applications, Volume 1*. Wiley, 3rd edition, January 1968.

[Ferreira *et al.*, 2008] Nivea De Carvalho Ferreira, Michael Fisher, and Wiebe Van Der Hoek. Specifying and reasoning about uncertain agents. *International Journal of Approximate Reasoning*, 49:35–51, 2008.

[Fine, 1972] Kit Fine. In so many possible worlds. *Notre Dame Journal of Formal Logic*, 13(4):516–520, 1972.

[Fitting, 1993] Melvin Fitting. Basic modal logic. In D.M. Gabbay, C.J Hogger, and J.A.Robinson, editors, *Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 1: Logical Foundations*. Oxford University Press, 1993.

[Friedman *et al.*, 1999] Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In *Proc. Sixteenth International Joint Conference on Artificial Intelligence (IJCAI '99)*, pages 1300–1307. Morgan Kaufmann, 1999.

[Gal and Pfeffer, 2008] Ya'akov Gal and Avi Pfeffer. Networks of influence diagrams: Reasoning about agents' beliefs and decision-making processes. *Journal of Artificial Intelligence*, 2008.

[Gmytrasiewicz and Doshi, 2005] Piotr J. Gmytrasiewicz and Prashant Doshi. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research*, 24:24–49, 2005.

[Goble, 1970] L.F. Goble. Grades of modality. *Logique et Analyse*, 13:323–334, 1970.

[Hajishirzi *et al.*, 2009] Hannaneh Hajishirzi, Afsaneh Shirazi, Jaesik Choi, and Eyal Amir. Greedy algorithms for sequential sensing decisions. In *Proc. Twenty First International Joint Conference on Artificial Intelligence (IJCAI '09)*, 2009.

[Halpern, 1990] Joseph Y. Halpern. An analysis of first-order logics of probability. *Artificial Intelligence*, 46(3):311–350, 1990.

[Hansen, 2004] Eric A. Hansen. Dynamic programming for partially observable stochastic games. In *IN PROCEEDINGS OF THE NINETEENTH NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 709–715, 2004.

[Harsanyi, 1967] John C. Harsanyi. Games with incomplete information played by "bayesian" players, IIII Part I. the basic model. *Management Science*, 14(3):159–182, 1967.

[Hart and Sharir, 1984] Sergiu Hart and Micha Sharir. Probabilistic temporal logics for finite and bounded models. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, STOC '84, pages 1–13, New York, NY, USA, 1984. ACM.

[Heifetz and Mongin, 1998] Aviad Heifetz and Philippe Mongin. The modal logic of probability. In *TARK '98: Proceedings of the 7th conference on Theoretical aspects of rationality and knowledge*, pages 175–185, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[Heifetz and Mongin, 2001] Aviad Heifetz and Philippe Mongin. Probability logic for type spaces. *Games and Economic Behavior*, 35:31–53, 2001.

[Hintikka, 1962] K. Jaako J. Hintikka. *Knowledge and Belief: An Introduction to the Logic of the Two Notions*. Cornell Press, Ithica, 1962.

[Jaeger, 1997] Manfred Jaeger. Relational Bayesian Networks. In *Proc. Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI '97)*, pages 266–273. Morgan Kaufmann, 1997.

[Kanger, 1957] S. Kanger. On the characterization of modalities. *Theoria*, 23:152–155, 1957.

[Kazakov and Pratt-Hartmann, 2009] Yevgeny Kazakov and Ian Pratt-Hartmann. A note on the complexity of the satisfiability problem for graded modal logics. In *Proceedings, Symposium on Logic in Computer Science*, pages 407–416, 2009.

[Kersting and Raedt, 2000] Kristian Kersting and Luc De Raedt. Bayesian logic programs. In J. Cussens and A. Frisch, editors, *Proceedings of the Work-in-Progress Track at the 10th International Conference on Inductive Logic Programming*, pages 138–155, 2000.

[Koller and Milch, 2001] Daphne Koller and Brian Milch. Structured models for multiagent interactions. In *In: Proceedings of the 8th conference on Theoretical Aspects of Rationality and Knowledge*, pages 233–248. Morgan Kaufmann Publishers Inc, 2001.

[Koller and Pfeffer, 1998] Daphne Koller and Avi Pfeffer. Probabilistic frame-based systems. In *Proc. National Conference on Artificial Intelligence (AAAI '98)*, pages 580–587. AAAI Press, 1998.

[Koller, 2001] Daphne Koller. Multi-agent influence diagrams for representing and solving games. In *Games and Economic Behavior*, pages 1027–1034, 2001.

[Kooi, 2003] Barteld P. Kooi. Probabilistic dynamic epistemic logic. *Journal of Logic, Language, and Information*, 12:381–408, 2003.

[Kozen, 1983] Dexter Kozen. A probabilistic pdl. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, STOC '83, pages 291–297, New York, NY, USA, 1983. ACM.

[Kozen, 1997] Dexter C. Kozen. *Automata and Computability*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1997.

[Kripke, 1963] Saul Kripke. Semantical considerations of model logic. *Zeitschrift fur Mathematishe Logik und Grundlagen der Mathematik*, 9:67–96, 1963.

[Lehmann and Shelah, 1983] Daniel J. Lehmann and Saharon Shelah. Reasoning with time and chance (extended abstract). In *Proceedings of the 10th Colloquium on Automata, Languages and Programming*, pages 445–457, London, UK, 1983. Springer-Verlag.

[McCarthy and Hayes, 1969] John McCarthy and Patrick J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969.

[Milch and Koller, 2000] Brian Milch and Daphne Koller. Probabilistic models for agents beliefs and decisions. In *Proceedings of the 16th conference on Uncertainty in Artificial Intelligence*, pages 389–396. Morgan Kaufmann, 2000.

[Milch and Koller, 2003] Brian Milch and Daphne Koller. Multi-agent influence diagrams for representing and solving games. *Games and Economic Behavior*, 45:181–221, 2003.

[Milch *et al.*, 2005] Brian Milch, Bhaskara Marthi, Stuart Russell, David Sontag, Daniel L. Ong, and Andrey Kolobov. Blog: Probabilistic models with unknown objects. In *Proc. Nineteenth International Joint Conference on Artificial Intelligence (IJCAI '05)*, pages 1352–1359. International Joint Conferences on Artificial Intelligence, 2005.

[Morgan and McIver, 1999] Carroll Morgan and Annabelle McIver. pGCL: formal reasoning for random algorithms. *South African Computer Journal*, 1999.

[Ng and Subramanian, 1992] Raymond T. Ng and V. S. Subramanian. Probabilistic logic programming. *Information and Computation*, 101(2):150–201, 1992.

[Ngo and Haddawy, 1995] Liem Ngo and Peter Haddawy. Probabilistic logic programming and Bayesian Networks. In *Asian Computer Science Conference*, pages 286–300, 1995.

[Nilsson, 1986] Nils J. Nilsson. Probabilistic logic. *Artif. Intell.*, 28(1):71–87, 1986.

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

[Pfeffer *et al.*, 1999] Avi Pfeffer, Daphne Koller, Brian Milch, and Ken T. Takusagawa. SPOOK: A system for probabilistic object-oriented knowledge representation. In *Proc. Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI '99)*, pages 541–550, 1999.

[Poole, 2003] David Poole. First-order probabilistic inference. In *Proc. Eighteenth International Joint Conference on Artificial Intelligence (IJCAI '03)*, pages 985–991, 2003.

[Reiter, 2001] Raymod Reiter. *Knowledge In Action: Logical Foundations for Describing and Implementing Dynamical Systems*. MIT Press, 2001.

[Richards and Amir, 2007] Mark Richards and Eyal Amir. Opponent modeling in scrabble. In *Proc. Twentieth International Joint Conference on Artificial Intelligence (IJCAI '07)*, page to appear. International Joint Conferences on Artificial Intelligence, 2007.

[Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov logic networks. *Mach. Learn.*, 62(1-2):107–136, 2006.

[Sack, 2009] Joshua Sack. Extending probabilistic dynamic epistemic logic. *Synthese*, 169(2):241–257, 2009.

[Shirazi and Amir, 2005] Afsaneh Shirazi and Eyal Amir. First order logical filtering. In *Proc. Nineteenth International Joint Conference on Artificial Intelligence (IJCAI '05)*, pages 589–595. International Joint Conferences on Artificial Intelligence, 2005.

[Shirazi and Amir, 2007] Afsaneh Shirazi and Eyal Amir. Probabilistic modal logic. In *Proc. National Conference on Artificial Intelligence (AAAI '07)*. AAAI Press, 2007.

[Shirazi and Amir, 2008] Afsaneh Shirazi and Eyal Amir. Factored models for probabilistic modal logic. In *Proc. National Conference on Artificial Intelligence (AAAI '08)*. AAAI Press, 2008.

[Shirazi and Amir, 2011] Afsaneh Shirazi and Eyal Amir. First-order logical filtering. *Artificial Intelligence Journal in Honor of John McCarthy*, 2011.

[Shirazi *et al.*, 2009] A. Shirazi, J. Hu, M. Singh, M. Squillante, and A. Mojsilovic. A framework for combined bayesian analysis and optimization for services delivery. 2009.

[Shoenfield, 1967] Joseph R. Shoenfield. *Mathematical Logic*. Addison-Wesley, Duke University, 1967.

[Tobies, 2000] Stephan Tobies. Probability logic for type spaces. *Journal of Logic and Computation*, 10:1–22, 2000.

[van Benthem *et al.*, 2009] Johan van Benthem, Jelle Gerbrandy, and Barteld Kooi. Dynamic update with probabilities. *Studia Logica*, 93(1):67–96, 2009.

[Wolter, 1999] Frank Wolter. First order common knowledge logics. *Studia Logica*, 65:249–271, 1999.

[Zettlemoyer *et al.*, 2008] Luke S. Zettlemoyer, Brian Milch, and Leslie Pack Kaelbling. Multi-agent filtering with infinitely nested beliefs, 2008.