A Complete Conditional Probability Logic and Its Dynamic Extensions

Mauricio Simões Camilo Hernandes
Informatics Department
SOKENDAI
(The Graduate University for Advanced Studies)

A thesis submitted for the degree of
Doctor of Philosophy

2015

Abstract

This thesis presents a system of axioms for a logic that is sound and complete with respect to conditional probability spaces.

The literature of *doxastic* logics (logics for beliefs) has a number of different ways of modeling beliefs of agents. One of the ways of expressing quantitative beliefs is through probability theory. Popper-Renyi conditional probability theory, an alternative approach to probability, is well suited for dynamic languages and able to express formulas of the form:

> 'I believe that p has probability of happening equal to 0 (zero). After p is announced to be true I believe in p with positive probability.'

In this thesis I define conditional probability spaces, and a system of axioms that is complete with respect to them. The proof of completeness for some extensions for dynamics is given by reduction axioms, another advantage of conditional probability logic.

As an application for this language I express strategies of card games. I use the card game Algo as an example and express game actions and strategies in a dynamic extension of the language of conditional probability logic.

To Tereza da Silva Hernandes (1927 - 2014)

Acknowledgements

I would like to thank my family, the biggest and best fund agency that has ever gave me support. Without them I would not have come this far.

I thank my advisor Makoto Kanazawa, who some key ideas in this thesis originated with. I am grateful for all his extensive and detailed comments that improved, both in content and in style, earlier versions of this text. His patience with my writings made them understandable. I thank him also for all these years in Japan teaching me matters way beyond logics.

My years in Japan were made much easier meeting so many wonderful people. Rita Kohl, Kaori Nagata and Marconi Madruga were specially important on the earlier stages after moving to Japan. I am not sure how far could I have gone without them. Thank you.

I would like also to thank Marie Seki, her patience to understand me made my later years in Japan worthwhile.

This thesis would not have been finished on time without the help of Flávia Dalmazo. Thank you for teaching me time management and so many other ninja moves.

The last weeks of writing were made easier with the help of Julio Assano and Thaís Finotto. Our long Skype meetings and chats were meaningful.

Finally, all the great people that I met through the last years either in Japan or outside. In one way or another you help me finish this thesis: Antonieta Pierotti, Mariet Paranuk, Andrea Zimova, Maha Sadi, Kaveh Maghsoudi, Claus Aranha, Yul Otani, Inbal Horev, Juliana Buritica, Felix Lee, Toshihiko Uchida, Carolina Alexiou, Marcos Ikegami, Erika Funa.

Contents

Chapter 1

Introduction

¡Oh! Y ahora, ¿quién podrá defenderme?

El Chapulín Colorado

This chapter is meant to help the reader not so familiar with logic understand at least some context which this text is talking to and about, and to help the reader more familiar with the subject to have a quick idea of what are the accomplishments of this thesis.

The first section is an informal introduction to the topics in the thesis. The second section uses some usual technical jargon to explain some of the ideas; it is meant to be understood by an initiate in symbolic logic studies. The theoretical background needed for the thesis is presented in the next chapter.

1.1 What is this thesis about?

This thesis is about logics to express probabilistic beliefs and their changes.

The idea of modeling beliefs with probabilities comes from the assumption that we can talk about degrees of beliefs. For instance, the belief that an apple falls after it is released from a tree, that it might rain tomorrow, that a flip of a coin will land heads or that a black box containing only red balls has a green ball (in this case the lack of belief). Those beliefs create different expectations, or as some say, you would be willing to bet different amounts of money on each of those events.

In cases where I am not sure, I can say, for example, that I believe that an event (say, that I will receive a Ph.D.) has probability 95% of happening. A logic of degrees of beliefs helps us to reason in a clear way about our own beliefs.

If I believe it is going to rain tomorrow and I believe that bringing an umbrella will keep me dry, then I should bring an umbrella. But we all see how often the weather forecast misses its predictions. My rule could be that I bring an umbrella only if it is going to rain with probability at least 80%; after all it is unpleasant to carry an umbrella that is unnecessary.

It is also interesting to note that beliefs can be wrong, as our earlier belief about the flatness of Earth was.

Either because our beliefs were wrong or because we were misinformed, we change our minds during our interactions with others. For example, if I have a biased coin I could flip it enough times and convince myself that it is more likely to land tails. A magician's performance is a good example of belief change. If I believed that a magician had cut his assistant in half only to see her appearing on the other side of the stage seconds after the box she entered was closed, then I could start considering the impossible a possibility.

Processes that lead to changes are numerous. However, we can put them in two different categories. First, the changes that occur only in our minds. This type of effect is often triggered by some announcement, either verbal or signaled in some other way. For example if you tell me you were born in Brazil, the probability that I assign to the event that you may speak Portuguese will increase.

Other changes modify the real world and consequently our beliefs. I believe I will be safer in a car if I fasten my seat belt, for instance. Changing the world changes our beliefs.

Also there are changes that are triggered by both at the same time.

Dynamics is an important theme in this thesis. Among other things, I propose a formal language to express changes when we are completely wrong in our beliefs and faced with true information that contradicts them. With this language we can express beliefs after events that were considered impossible (that had probability zero) happen.

Finally, this thesis applies the language presented to card games. Games are a rich environment to describe with logic because of the intrinsic flow of information and changes that they present. The focus is exclusively on a game called Algo, a card game created by the Japanese Olympic Arithmetics Committee.

I sum up the accomplishments of the present work as follows:

1. The design of a language able to express conditional probabilistic beliefs;

2. The introduction of a language with dynamic operators to express flow of information (announcements, events and assignments);

3. The study of the card game Algo, expressing strategies with the logical language established in previous chapters.

1.2 What are the contributions of this thesis?

1.2.1 Conditional Probability Logic

This thesis presents a system of axioms for a logic that is sound and complete with respect to conditional probability spaces.

As we will see, the literature of *doxastic* logics (logics for beliefs) has a number of different ways of modeling beliefs of agents. One of the ways of expressing quantitative beliefs is through probability theory. The traditional Kolmogorov probability theory is not rich enough to handle languages with dynamic operators (see Chapter 4).

Renyi conditional probability theory, an alternative approach to probability, is well suited for dynamic languages. In Rényi (1955) the conditional probability function $P(\cdot \mid \cdot)$ is defined as $P : \Sigma \times \mathcal{B} \to [0, 1]$ satisfying:

(1) $\mu(A \mid B) \geq 0$; further $\mu(B \mid B) = 1$;

(2) For any fixed $B \in \mathcal{B}$ and for any countable sequence $(A_i)_{i\in\mathbb{N}}$ of pairwise disjoint elements of Σ it holds that $\mu(\bigcup_i^\infty A_i, B) = \sum_i^\infty \mu(A_i, B)$ (we call this property σ-additivity);

(3) $\mu(A \cap B \mid C) = \mu(A \mid B \cap C) \cdot \mu(B \mid C)$;

where Σ is a σ-algebra over some set Ω and $\mathcal{B} \subset \Sigma \setminus \{\varnothing\}$. Note that we can define the standard probability function as $P(A) = P(A \mid \Omega)$.

In the first chapter of this thesis I define conditional probability spaces, and the system of axioms that is complete with respect to them.

To express beliefs of probabilities I will use a formula of the form $L_r p$ meaning 'the agent believes that the probability of p being true is at Least r.' This language was proposed by Aumann (1999) who did not prove completeness. Heifetz and Mongin (2001) proposed first a set of axioms that was complete with respect to probability spaces. In Zhou (2009) a simpler set of axioms was proposed.

The first contribution of this thesis is an extension of the language from Aumann to express conditional probabilities. I define the connective $L_r(p||q)$ with the intended meaning 'the agent believes that the probability of p given q is at Least r.'

The relevance of such a language is not straightforward. I spend the rest of this section explaining how a language for conditional probability belief is relevant. A similar discussion was given in Baltag and Smets (2008).

Example 1.1 (Lottery Paradox)**.** Suppose that, for some $r < 1$, $L_r p$ implies 'The agent believes p.' Pick a natural number n such that $r \leq \frac{n-1}{n}$.

Suppose there is a lottery with n tickets, and the agent has one of the tickets. Assuming the lottery is fair, her chances of winning are $1/n$. Therefore the agent believes that her ticket will not be the winning ticket (chances of not winning is $\frac{n-1}{n}$). In fact, for any given ticket the agent believes that it will not be the winning ticket. Hence, she believes that no ticket will be the winning ticket. Which is a contradiction since she knows that one of them will be the winning one.

This example makes clear the idea that qualitative belief in a proposition p should imply quantitative belief that the chance of p is 1.

On the other hand, an important characteristic of belief is that it can be wrong, i.e., an agent can believe in an event p (believe its chance is 1) but in fact it is false. On dynamic scenarios for probabilistic beliefs where updates are calculated by Bayesian rule it is reasonable to expect events with probability zero. However, the Bayesian update is not defined for events with probability zero.

The notion of conditional probability belief, defined in this thesis, allows a natural way to express probabilistic beliefs integrated with dynamic scenarios.

1.2.2 Solving Open Problems

Baltag and Smets (2008) define a language for *qualitative* conditional belief, i.e., an operator $L_r(p||q)$ with $r \in \{0, 1\}$. Roughly, this operator means 'the agent believes in p given q' if $r = 1$; and 'the agent does not believe in p given q' if $r = 0$.[1]

This qualitative language is complete with respect to finite discrete conditional probability spaces, i.e., in the definition of conditional probability function the σ-algebra Σ is finite and equal to the set of subsets of Ω ($\Sigma = \mathcal{P}(\Omega)$, and Ω is finite).

[1] Baltag and Smets (2008) does not mention the number r, but it is useful to denote with the subscript r for easier comparison with the present work.

The conditional probability logic defined in Chapter 3 is a language for *quantitative* conditional belief. The qualitative language in Baltag and Smets (2008) can be seen as a sublanguage of the conditional probability logic. Conditional probability logic is complete with respect to the class of conditional probability spaces[1] with $\mathcal{B} = \Sigma$.

It is worth mentioning that in the finite discrete case, Renyi's definition and Popper's definition of Conditional Probability coincide, as commented by Baltag and Smets (2008). However, in this thesis they are essentially different.

To sum up this part I restate two problems that are left open in Baltag and Smets (2008) that this thesis addresses:

Open Problem 1. To axiomatize the logic for arbitrary (finite or infinite) conditional probability models.

Open Problem 2. Study the logic obtained by adding quantitative modal operator.

1.2.3 Dynamics and Reduction Axioms

The major advantage of conditional probability theory as the underlining model for a language lies on the dynamics side. Announcements of propositions with probability zero can be made and after the announcement the agent believes in the proposition with positive probability. The intuition behind this problem is easy to grasp.

Imagine a shelf with three bananas, one apple and one screwdriver. Note that the following should be equivalent in a logic of announcements (a traditional dynamic logic):

(a) After you take an object from the shelf and tell me it is a fruit I should assign probability 1/4 to the fact that you took an apple; and

(b) I assign probability 1/4 to the event of you taking an apple given you took a fruit.

There are languages (based on the traditional Kolmogorov probability) that are able to express this situation. But the problem lies in the following similar scenario:

Imagine the same shelf as before with three bananas, one apple, one screwdriver and one knife, but I do not know about the knife. The following should also be equivalent in a logic of announcements:

(a) After you take an object from the shelf and tells me it is a cooking tool I should assign positive probability to the fact that you took a knife; and

[1]I use a modified definition of conditional probability function. In Renyi's definition $\mathcal{B} \subset \Sigma \setminus \{\varnothing\}$, whereas in Chapter 2 I define conditional probability function slightly differently; see Definition 2.1.

(b) I assign positive probability to the event that you will take a knife given you took a cooking tool.

Let $![p]$ be the operator for the public announcement of p and say that a formula of the form $[!p]q$ means 'after the announcement of p it is the case that q.' In Chapter 4 we define this operator formally.

In order to prove completeness of Public Announcement Logic we find *reduction axioms*, a standard technique. The idea behind reduction axioms is: given a formula ϕ with the new operator $[!\cdot]$ (say $[!p]q$) define a recursive method to find an equivalent formula without the new operator $[!\cdot]$ (in this case $p \to q$).

The point to keep in mind at this stage is the importance of conditional probability logic. After a public announcement a formula without conditional part is equivalent to a formula with conditional part, i.e., we have the following equivalence as a theorem:

$$[!p]L_r q \leftrightarrow L_r([!p]q||p).$$

Conditional probability logic is essential for the existence of reduction axioms.

Dynamic epistemic logic is well studied in the book van Ditmarsch et al. (2007), however it does not treat the probabilistic case. The papers of van Benthem (2003), Sack (2009) and then Kooi (2003) were the first to treat the probabilistic case. The notion of uncertainty of an announcement was improved in van Benthem et al. (2009). However, all these references were based on the Kolmogorov notion of probability and used the static language proposed by Fagin and Halpern (1994), which is a more expressive language than the language found in Zhou (2009).

This thesis presents a language for conditional probability logic, which is essential to express the reduction axioms of some dynamic logics (like Public Announcement Logic) in the spirit of Aumann (1999) and Zhou (2009). Later in this thesis we come back to the distinction between Aumann's and Halpern's ideas.

1.2.4 Card Games

As an application of this language I express strategies of card games. I use the card game Algo as an example and express card actions and strategies in a dynamic extension of the language of conditional probability logic.

Chapter 2

Background

Só escreva quando de todo não puder
deixar de fazê-lo. E sempre se pode.

Carlos Drummond de Andrade

Just write when you cannot lay it
aside. And one always can.

Carlos Drummond de Andrade

We present in this chapter the theoretical background needed for this thesis. Since all the definitions and results are well known in the literature (with exception of conditional probability measure) we present them without examples to motivate their point.

2.1 Set Theory and Propositional Logic: Basic Notation

First, we briefly define some notation of set theory and syntax for logical languages.

Propositional logic is the most popular as the base language for epistemic logics. We define propositional logic with the symbols $\neg, \wedge$ and $\vee$ which denote negation, conjunction and disjunction. The symbol $\rightarrow$ denotes implication. The set P is a set of countable propositional variables. Any propositional variable is a formula; if ϕ and ψ are formulas so are $\neg\phi$, $\phi \wedge \psi$, $\phi \vee \psi$ and $\phi \rightarrow \psi$; and nothing else is a formula. A more elegant way to express it is:

A formula ϕ is given by

$$\phi \doteq p|\neg\phi|\phi \wedge \phi|\phi \vee \phi|\phi \to \phi$$

where $p \in P$.

The notation and sets of set theory are denoted as follows. The sets of natural and rational numbers are denoted by $\mathbb{N}$ and $\mathbb{Q}$. The symbols $\cap$ and $\cup$ denote intersection and union of sets. The symbol $\subset$ denotes inclusion. If X is a set, then $\mathcal{P}(X)$ denote the power set of X, i.e., the set of all subsets of X.

2.2 Measure and Conditional Probability

In this section we define the basic elements of measure theory and conditional probability that we will make use of throughout this thesis. For more details in measure theory see, e.g., Halmos (1950). Our definition of conditional probability function is a modified version of Rényi (1955). In his original proposal the conditional probability function was defined as $P : \Sigma \times \mathcal{B} \to [0, 1]$, where $\mathcal{B}$ is closed by countable disjoint union and $\varnothing \notin \mathcal{B} \subset \Sigma$. In this thesis we consider the case that $\mathcal{B} = \Sigma$, with the added assumption that $P(\cdot, \varnothing)$ is not σ-additive.

Fix a set X. $\Sigma \subset \mathcal{P}(X)$ is a *σ-algebra* if

· $\varnothing, X \in \Sigma$,

· if $A, B \in \Sigma$, then $A \setminus B \in \Sigma$ and

· if $A_n \in \Sigma$ for all $n \in \mathbb{N}$, then $\bigcup_{n\in\mathbb{N}} A_n \in \Sigma$.

Call (X, Σ) a *measurable space.*

We say that a measurable space (X, Σ) is *finite* if X is finite. And we say it is *discrete* if $\Sigma = \mathcal{P}(X)$.

A function $\mu : \Sigma \to [0, 1]$ is called *σ-additive* if for any countable family of pairwise disjoint sets $(A_i)_{i\in\mathbb{N}}$ in Σ, it holds that $\mu(\bigcup_{i\in\mathbb{N}} A_i) = \sum_{i=0}^{\infty} \mu(A_i)$.

Let (X, Σ) be a measurable space, a function $\mu : \Sigma \to [0, 1]$ is called a *probabilistic measure (function)* if the following is satisfied, for any $A \in \Sigma$:

(1) $\mu(A) \geq 0$;

(2) μ is σ-additive;

(3) $\mu(X) = 1$.

The triple (X, Σ, μ) is called a *probability space.*

Definition 2.1. Let (X, Σ) be a measurable space and denote by Σ^* the set $\Sigma \setminus \{\varnothing\}$. A function $\mu : \Sigma \times \Sigma \to [0, 1]$ is called a *conditional probabilistic measure (function)* if the following is satisfied:

(1) $\mu(A, B) \geq 0$; further $\mu(B, B) = 1$;

(2) For any fixed $B \in \Sigma^*$ $\mu(\cdot, B)$ is σ-additive;

(3) $\mu(A \cap B, C) = \mu(A, B \cap C) \cdot \mu(B, C)$.

The triple (X, Σ, μ) is called a *conditional probability space.*

The traditional probability of an event A is defined as $\mu(A|X)$. It is useful to state and prove the following properties of a conditional probability function:

Theorem 2.2. Let (X, Σ, μ) be a conditional probability space and let A, B, C be elements of Σ. The following hold:

1. $\mu(A, \varnothing) = 1$;
2. $\mu(A \cap B, B) = \mu(A, B)$;
3. If $A \subseteq B$, then $\mu(B, A) = 1$;
4. $\mu(A \cap B, A \cap C) = \mu(B, A \cap C.)$

Proof. (1) Note that with $B = C = \varnothing$ in (3) of Definition 2.1 we have $\mu(\varnothing, \varnothing) = \mu(A, \varnothing) \cdot \mu(\varnothing, \varnothing)$, hence $\mu(A, \varnothing) = 1$ by (1) of Definition 2.1.

(2) Letting $B = C$ in (3) of Definition 2.1 we have

$$\mu(A \cap B, B) = \mu(A, B) \cdot \mu(A, A) = \mu(A, B).$$

(3) Remember that if $A \subseteq B$, then $A \cap B = A$. The proof is given as follows:

$$1 = \mu(B \cap A, A) = \mu(B, A) \cdot \mu(A, A) = \mu(B, A).$$

(4) Note that $A \cap B \cap C \subseteq A$. Then by item (3) $\mu(A, A \cap B \cap C) = 1$. With that the proof can be seen by the following:

$$\mu(A \cap B, A \cap C) = \mu(A, A \cap B \cap C) \cdot \mu(B, A \cap C) = \mu(B, A \cap C).$$

□

Let X be a set and $(B_i)_{i \in I}$ a family of subsets of X indexed by the nonempty set I. Denote by $\sigma\{B_i : i \in I\}$ the smallest σ-algebra containing the family $(B_i)_{i \in I}$.

Let (X, Σ_1) be a measurable space and (Y, Σ_2) a finite discrete measurable space. Let μ and η be measures over Σ_1 and Σ_2. The *product space* between these two spaces is given by $(X \times Y, \Sigma_1 \times \Sigma_2)$ where $\Sigma_1 \otimes \Sigma_2 = \sigma(\{E \times F : E \in \Sigma_1, F \in \Sigma_2\})$, the smallest σ-algebra containing $\Sigma_1 \times \Sigma_2$. The *product measure* between μ and η is denoted by $\mu \times \eta$:

$$\mu \times \eta(U) = \sum_{y \in Y} \mu(U_y)\eta(y),$$

where $U_y = \{x \in X : (x, y) \in U\}$, and $\eta(y) = \eta(\{y\})$. It is a well-known fact (see, e.g., Halmos (1950)) that in the case where μ and η are probability measures the product $\mu \times \eta$ is a probability measure.

A real valued function $f : X \to [0, 1]$ is a Σ-*measurable function* if for all r between 0 and 1 we have $f^{-1}([0, r]) \in \Sigma$. Or, equivalently, $f^{-1}([r, 1]) \in \Sigma$ for all $r \in [0, 1]$.

A function $T : X \times \Sigma \to [0, 1]$ is a *Markov kernel* if for any $x \in X$ the function $T(x, \cdot)$ is a probabilistic measure over Σ and for any $A \in \Sigma$ the function $T(\cdot, A)$ is a Σ-measurable function.

A function $T : X \times \Sigma \times \Sigma \to [0, 1]$ is a *conditional Markov kernel* if for any $x \in X$ the function $T(x, \cdot, \cdot)$ is a conditional probabilistic measure over Σ and for any $A, B \in \Sigma$ the function $T(\cdot, A, B)$ is a Σ-measurable function.

The probability theory for conditional probability spaces was proposed in Rényi (1955). In his original proposal the conditional probability function was defined as $P : \Sigma \times \mathcal{B} \to [0, 1]$, where $\mathcal{B}$ is closed under countable disjoint union and $\varnothing \notin \mathcal{B} \subset \Sigma$. In this thesis we consider the case that $\mathcal{B} = \Sigma$, with the added assumption that $P(\cdot, \varnothing)$ is not σ-additive.

Game theorists adopted this approach some time later in Myerson (1986) and more recently in Battigalli and Siniscalchi (1999). The logic community seems to have not paid much attention until Baltag and Smets (2008), which proposed a language for finite discrete spaces. We will come back to this question in the next chapter.

2.3 Epistemic Structure and Probabilistic Models

An *epistemic structure* is a tuple $\mathcal{M} = \langle W, R, v\rangle$ such that W is a set, R is an equivalence relation over the power set of W, and v is a valuation, i.e., a function from the set of propositions to a subset of W.

A *probability model* $\mathcal{M} = \langle \Omega, \Sigma, T, v\rangle$ is such that (Ω, Σ) is a measurable space; $T : \Omega \times \Sigma \to [0, 1]$ is a Markov kernel; v is a valuation, a function from the set of propositional variables to Σ.

Definition 2.3. A *conditional probability model* $\mathcal{M} = \langle \Omega, \Sigma, T, v \rangle$ is such that (Ω, Σ) is a measurable space; $T : \Omega \times \Sigma \times \Sigma \to [0, 1]$ is a conditional Markov kernel; v is a valuation, a function from the set of propositional variables to Σ.

2.4 Syntax - Halpern and Aumann

In this thesis we will be referring to two formal languages for probability. Here we call them Halpern's language and Aumann's language; however this distinction is usually not made as texts in general use only one language and stick to it. For clarity I make this distinction here. The names are from the first person to propose each language.

Halpern's language is given by:

$$\phi \doteq p|\neg\phi|\phi \wedge \phi|\alpha_1 \cdot P(\phi) + ... + \alpha_k \cdot P(\phi) \geq \beta,$$

where $p \in P$ and $\alpha_1, ..., \alpha_k, \beta \in [0, 1] \cap \mathbb{Q}$.

$P(\phi)$ stands for the probability of a formula ϕ. This language allows for combination of probabilities of formulas in a linear way, e.g, the formula $P(\phi) + P(\psi) \geq 0.5$ is a formula of the language.

The disadvantage of this language is that its axiomatization needs a set of axioms for inequalities, turning the proof of completeness less traditional.

Aumann's language is, strictly speaking, a sublanguage of Halpern's. It does not allow linear combination of probabilities and is given by:

$$\phi \doteq p|\neg\phi|\phi \wedge \phi|P(\phi) \geq r,$$

where $r \in [0, 1] \cap \mathbb{Q}$.

It is standard to write $P(\phi) \geq r$ as $L_r\phi$ with the same intended meaning 'the probability of ϕ is at least r.'

The disadvantage of this language is its inability to express sums of probabilities. As a consequence the dynamic extensions are harder to study over Aumann's language. In Chapter 4 we show how conditional probability logic is needed in order to express some dynamic concepts within Aumann's idea. However, completeness is achieved only for a limited class of event models.

Admittedly the disadvantages of either Halpern's or Aumann's language could be ignored if the completeness of a logic is the only interest, since Halpern's language is simply more expressive and also complete. What I argue here is that despite the fact that Aumann's language is less expressive it is still interesting as an object of study for two reasons. First, the techniques for the proof of completeness (see Chapter 2) are similar to modal logics, i.e., L_r operator behaves in many ways similarly to modal logics (it is a modal operator in spirit). Second, it is still possible to express useful concepts despite the loss of expressivity (like some strategies in Chapter 4). Moreover, the conditional probability language (that is an extension of Aumann's language) presented in this thesis was designed with dynamics in mind, which allowed simple formulas for non-static scenarios (see Chapter 3).

Intuitively we can see this simplicity as a consequence of the following difference: On Kolmogorov theory of probability we say 'The conditional probability of A given B is $P(A \cap B)/P(B)$ if $P(B) \neq \varnothing$ and is undefined otherwise'. On Popper-Renyi theory of conditional probability (Definition 2.1) we say 'The conditional probability of A given B is $P(A|B)$' - the term $P(A|B)$ is always defined.

2.5 Dynamic epistemic logic

Dynamic epistemic Logic is a field well studied, e.g., van Ditmarsch et al. (2007). Probabilistic dynamics as an extension of Halpern's language is studied in van Benthem (2003), Kooi (2003), and Sack (2009). We illustrate dynamics in this section with public announcement logic. For more details see Chapter 4.

The announcement of a formula ϕ is denoted by $[!\phi]$.

$[!\phi]\psi$ is read as 'after the announcement of ϕ, it is true that ψ.'

It is interesting to note that there are different ways of defining the updated model in probability logics for the case when the announced formula has probability zero (ϕ is such that $L_1\neg\phi$ holds). Below we define one possible way of defining such update.

Definition 2.4 (SatisfiabilitySack (2009))**.** If $\mathcal{M}$ is a probabilistic model, the updated model after the announcement of ϕ is denoted by $\mathcal{M}_\phi$ and defined as

$$\mathcal{M}_\phi = \langle \Omega_\phi, \Sigma_\phi, T_\phi, v_\phi \rangle,$$

where

$\Omega_\phi = [\![\phi]\!]$;

$$\Sigma_\phi = \{B \cap \Omega_\phi : B \in \Sigma\};$$
$$T_\phi(w, A) = \begin{cases} \dfrac{T(w, A \cap [\![\phi]\!])}{T(w, [\![\phi]\!])} & \text{if } T(w, [\![\phi]\!]) \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$
$$v_\phi(\cdot) = v(\cdot) \cap \Omega_\phi$$

In Sack (2009) there is a set of reduction axioms for public announcement logic, but note that

$$L_0 p \wedge p \wedge [!p] L_r p$$

holds only if $r = 0$.

2.6 Universal Modality

Let $\mathcal{M} = \langle W, \Sigma, T, v \rangle$ be a conditional probability model. Let p be a propositional variable and denote by $[\![p]\!]$ the set $\{w \in W : w \in v(p)\}$. Note that by the definition of conditional probability function (Definition 2.1), for any $x \in W$, if $[\![p]\!] \neq \varnothing$, then $T(x, \cdot, [\![p]\!])$ is σ-additive.

Suppose we have a language in which the set of formulas Φ is satisfied by $\mathcal{M}$ (satisfied in each state w in $\mathcal{M}$) if and only if $T(x, \cdot, [\![p]\!])$ is σ-additive. Suppose the connective $\Diamond$ is such that $\Diamond p$ is true if and only if $[\![p]\!] \neq \varnothing$. Then we can define the set $\Phi_\Diamond = \{\Diamond p \to \phi : \phi \in \Phi\}$, which holds if and only if $[\![p]\!] \neq \varnothing$ implies $T(x, \cdot, [\![p]\!])$ is σ-additive.

Informally we translated part of the conditional probability function definition to a formal language. I will spell out all the details in Chapter 3. For now, let's define the $\Diamond$ operator which is the dual of the traditional *universal* modality.

The $\Box$ connective - with the traditional intended meaning '$\Box\phi$ holds in a state if and only if ϕ holds at all states in the model' - is the dual of the desired connective ($\Diamond$) above. We abbreviate $\neg\Box\neg A$ by $\Diamond A$ and from now on we use the $\Box$ as the primitive connective following Goranko and Passy (1992), an interesting discussion on the universal modality. A nice introduction to the topic can be found in Blackburn et al. (2002).

2.7 Algo

Card games, due to their incomplete information scenario (some cards are hidden in other players' hand), are a rich field for logicians and computer scientists to apply and test their models. For instance, Billings et al. (2002) discussed an automated player for poker. On the other hand, van Ditmarsch (2001) modeled a game called *Cluedo* in terms of the players' knowledge states and how they change after public announcements are made and private information is shared.

In this thesis I use the language defined for probabilities and conditional probabilities to express the epistemic states and actions of card games, focusing on a game called Algo.[1] Created by the Japanese Arithmetics Olympic Committee, this game has not been well studied from the mathematical or logical point of view and it is particularly interesting from the point of view of epistemic logic.

Another contribution of this work is to propose some strategies for an automated player (AI) based on the epistemic models defined in this paper.

For reference we explain the rules of the game below:

Definition 2.5 (Algo)**.** $Algo_n$ has the following rules:

> The game has two players; there are n black cards and n white cards (with the back and the front with the same color), and both sets of cards are labelled from 0 (zero) to $n-1$. All the cards are shuffled and both players receive four cards, the rest of the cards are placed face down in a pile. Each player orders his/her own cards, with the smallest card on the left, and place them face down in front of them; if a black card and a white card have the same number, the black card is considered smaller in this ordering.
>
> The randomly chosen first player takes a card from the pile, looks at its value and *attacks* the opponent's cards. An attack consists in guessing the value of a card in the opponent's hand: 1. if the player guesses correctly, the card is turned face up and the attacker can choose either to attack again or to stop; when the attacker stops attacking, he/she inserts the card just taken from the pile in the correct position in his hand, face down; 2. if the player's guess is wrong, he/she must place the card taken from the pile with his/her other cards in order and turn it face up. If the attacking player stops or guesses incorrectly it is the other player's turn to take a card from the pile and attack.
>
> When only one player has any card face down, the game ends and that player is the winner.

[1] This game is also known as "Coda."

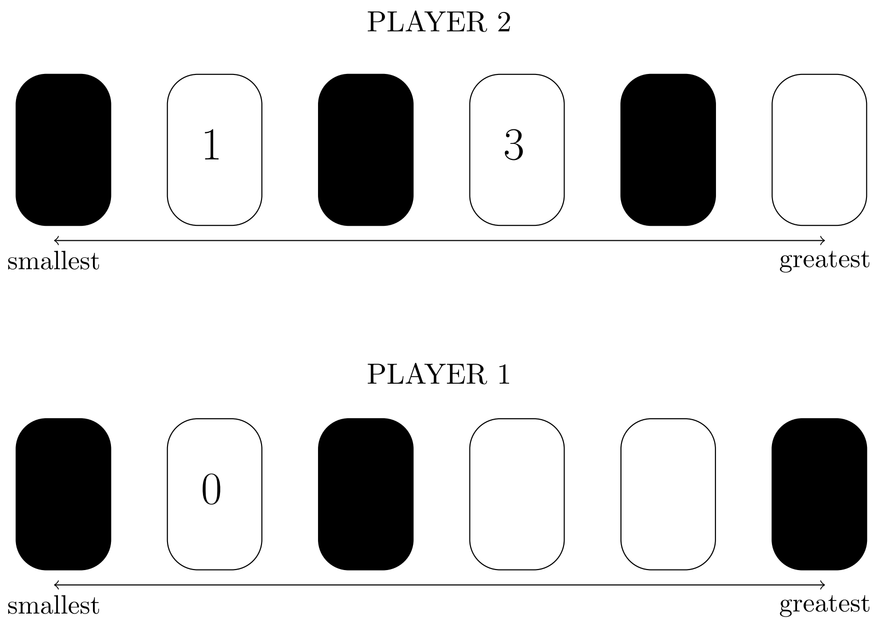

Figure 2.1: Example 2.6. Example of a state go Algo. Player 1 has the cards = $(0B, 0W, 2B, 2W, 4W, 5B)$. Player 2 has the cards = $(1B, 1W, 3B, 3W, 4B, 5W)$

The following example is question 7 in Japanese Arithmetics Olympic Committee (2011).

Example 2.6. An example of a moment of $Algo_6$ can be seen in Figure 2.1. A property of such a state is that we can completely determine all the cards in both players' hands only by looking at the information available:

In this case the value of the cards in the hand of player 1 is $(0, 0, 2, 2, 4, 5)$ and $(1, 1, 3, 3, 4, 5)$ for player 2.

The standard game of Algo has 24 cards (12 black and 12 white) from 0 (zero) to 11, so when referring to $Algo_{12}$ we may drop the subscript.

Algo Language When stating examples for $Algo_n$ we will make use of the following language.

Let the set of propositions be $P_c = \{vcp_i : v < n, c \in \{B, W\}, p$ is a player, $i \leq n\}$, v stands for the value of the card, c stands for the color, p stands for the player holding the color and i stands for the position in the hand of the player. Note that Algo has

$2n$ cards but any player holds up to n cards in any given situation because each player draws one card from the pile at each turn and if the pile is empty they stop drawing.

Fix also the following set of propositions $P_f = \{U_{p_i}, D_{p_i} : i \leq n, p \text{ is a player}\}$, where U_{p_i} stands for 'card in position i of player p is face up' and D_{p_i} stands for 'card in position i of player p is face down.' The language for Algo is given by:

$$\phi \doteq p|\neg\phi|\phi \wedge \phi,$$

where $p \in P_c \cup P_f$.

Example 2.7. *(Ex.2.6 continued)* The previous example can be stated with our formal language. Below we state only the propositions without negation:

Player 1 $= \{0B1_1, 0W1_2, 2B1_3, 2W1_4, 4W1_5, 5B1_6\} \cup \{U_{1_2}\}$

Player 2 $= \{1B2_1, 1W2_2, 3B2_3, 3W2_4, 4B2_5, 5W2_6\} \cup \{U_{2_2}, U_{2_4}\}$

Guess Notation One of the ways in which information flows in Algo is through attacks (or guesses). Later when defining our formal language for Algo we intend to express formulas like 'after player 1 guesses that the opponent has a 2B, player 2 believes that player 1 has the card 3B with probability 0.5.' With this in mind we define the following notation for a guess:

$$g : Pa \xrightarrow{vcj_i} Pj$$

Where Pa is the attacker, vcj_i is the guess and Pj is the player receiving the attack. Note that Pj is redundant, but we leave it as it is.

The guess 'Player 1 guesses that player 2 has the 2B in position 2' is denoted as $g : P1 \xrightarrow{2B2_2} P2$.

Chapter 3

Conditional Probability Logics

> Que ninguém se engane, só se consegue a simplicidade através de muito trabalho.
>
> Clarice Lispector

> That no one be mistaken, one reaches simplicity only through a lot of work.
>
> Clarice Lispector

3.1 Introduction

In this chapter we introduce conditional probability logic (CPL). This language sets the foundation for the study of dynamics in later chapters. Dynamics in epistemic logic is an important field of study for artificial intelligence, philosophy and mathematics. While non-probabilistic languages are well understood van Ditmarsch (2001), the same is not true for languages able to express probabilistic beliefs; see for example Baltag and Smets (2008)[1].

The syntax in this chapter is an extension of previous probabilistic languages. We extend the logic presented in Heifetz and Mongin (2001) and simplified in Zhou (2009),

[1] As we mentioned in the introduction, Baltag and Smets (2008) proposes a language for finite discrete conditional probability spaces which expresses qualitative beliefs.

where the set of axioms proposed was sound and complete for the class of type spaces introduced by Harsanyi (1967).

Using Halpern's approach to probability logic in a previous work, Fagin et al. (1990) discusses the possibility of expressing conditional probabilities extending its language to express product, e.g., they would express a formula of the form $P(A|B) \geq \frac{1}{2}$ by a formula of the form $P(A \cap B) \geq 2P(B)$. However expressions of the form $P(A|B) + P(B|A) \geq \frac{1}{2}$ would lead to nonlinear combination of terms. This led to an increase in complexity of the language. Moreover one can see that this syntax is still based on the traditional probability theory of Kolmogorov. More recently, extending Halpern's language, van Benthem (2003) added dynamics with conditional probabilities but the underlying probability theory was still the one defined by Kolmogorov.

Although the probability theory for conditional probability spaces was well described in Rényi (1955) and game theorists adopted this approach some time later in Myerson (1986) and more recently in Battigalli and Siniscalchi (1999), the logic community has not fully adopted this approach in its languages yet.

In this chapter we extend the language of Zhou (2009) to a system able to express conditional probability. Furthermore, we propose the class of conditional probability spaces as semantics. Interestingly enough, Aumann's perspective on probability logics had no conditional probability extensions until this work. Moreover, neither Aumann's nor Halpern's languages have any other proposal for conditional probability language with quantitative degrees of beliefs.

As mentioned in the introduction, in this chapter we address two open problems proposed by Baltag and Smets (2008):

Open Problem 1. To axiomatize the logic for arbitrary (finite or infinite) conditional probability models.

Open Problem 2. Study the logic obtained by adding conditional belief operator.

To start the discussion we take the following problem as a motivation:

Problem 3.1. Let $[!p]$ stand for the announcement of a proposition p and let $L_r p$ stand for 'agent believes that the probability of p being the case is at least r'. How to define a logical system (a set of axioms) where the formula $p \wedge (L_1 \neg p) \wedge [!p] L_r p$ is consistent for some $r > 0$? What should a model that satisfies such a formula be like?

The goal of this chapter is to axiomatize the logic of belief operator for possibly infinite conditional probability models, i.e., to establish a set of axioms sound and complete with respect to the class of conditional probability models (Definition 2.3). Traditionally, belief operators[1] describe the beliefs of agents. The idea of employing probability languages for beliefs is to express degrees of beliefs. For example, $L_{0.5}p$ for expressing 'the agent's beliefs in p is at least 0.5.' The motivation behind conditional probability logic is to express formulas of the form $L_r(p||q)$ with the intended meaning being 'the agent's belief in p given q is at least r.'

When referring to probabilities we refer to subjective probabilities. A logic expressing objective probability should have an axiom of the form $L_1 p \to p$ (meaning 'if p has probability 1, then p is true'). However Heifetz and Mongin (2001) mentions that:

> We do know not if it is possible to formulate a general semantic counterpart for truth axiom schema of epistemic logic $L_1\phi \to \phi$.

Probability spaces (and conditional probability spaces) are structures appropriate to reason about beliefs, but more work is needed to reason about objective probabilities.

3.2 Language and Model

We define *conditional probability language* as follows. Let $P = \{p, q, ...\}$ be the countable set of propositional variables and consider the language obtained from the following:

$$\phi \doteq \top | p | \neg\phi | \phi \wedge \phi | \Box\phi | L_r(\phi||\phi),$$

where $p \in P$ and $r \in [0,1] \cap \mathbb{Q}$.[2] We use the following abbreviations:

$$\bot \doteq \neg\top$$

$$M_r(\phi||\psi) \doteq L_{1-r}(\neg\phi||\psi)$$

$$L_r\phi \doteq L_r(\phi||\top)$$

$$\Diamond\phi \doteq \neg\Box\neg\phi$$

$$\phi \vee \psi \doteq \neg(\neg\phi \wedge \neg\psi)$$

$$\phi \to \psi \doteq \neg\phi \vee \psi.$$

[1] For instance Bp, meaning 'the agent believes that p.'
[2] $\mathbb{Q}$ is the set of rational numbers.

The formula $M_r\phi$ has the intended meaning 'the agent believes that the probability of ϕ is at most r.' The connective $\Box$ is the universal modality. A formula of the form $\Box\phi$ has the intended meaning 'ϕ holds in every state.'

Denote by $\mathcal{L}$ the set of all formulas in this language. Note that $\mathcal{L}$ is countable, assuming that the set of propositional variables P is countable.

Definition 3.2. Consider a conditional probability model (CPM) $\mathcal{M} = \langle \Omega, \Sigma, T, v \rangle$ and define the relation $\models$ over $\mathcal{M}$ as follows:

$\mathcal{M}, w \models p$ iff $w \in V(p)$;
$\mathcal{M}, w \models \neg\phi$ iff $\mathcal{M}, w \not\models \phi$;
$\mathcal{M}, w \models \phi \wedge \psi$ iff $\mathcal{M}, w \models \phi$ and $\mathcal{M}, w \models \psi$;
$\mathcal{M}, w \models \Box\phi$ iff $\mathcal{M}, w' \models \phi$ for all $w' \in W$;
$\mathcal{M}, w \models L_r(\phi||\psi)$ iff $T(w, [\![\phi]\!], [\![\psi]\!]) \geq r$;
where $[\![\phi]\!] = \{w \in \Omega : \mathcal{M}, w \models \phi\}$.

Lemma 3.3. For any $\phi \in \mathcal{L}$, $[\![\phi]\!] \in \Sigma$.

Proof. The proof is by the complexity of ϕ, where the atomic case is given by the definition of the valuation v. The propositional case is given by the fact that $[\![\neg\phi]\!] = \Omega \setminus [\![\phi]\!]$ and $[\![\phi_1 \vee \phi_2]\!] = [\![\phi_1]\!] \cup [\![\phi_2]\!]$. For the universal case just note that $[\![\Box\phi]\!] = \Omega$ or $\varnothing$. The case of formulas of the form $L_r(\phi||\psi)$ is given by noticing that if $[\![\phi]\!], [\![\psi]\!] \in \Sigma$, then $[\![L_r(\phi||\psi)]\!] = \{w \in \Omega : T(w, [\![\phi]\!], [\![\psi]\!]) \geq r\} = T^{-1}(\cdot, [\![\phi]\!], [\![\psi]\!])([r, 1])$, and note that $T(\cdot, [\![\phi]\!], [\![\psi]\!])$ is a measurable function. □

A formula ϕ is said to be *true* in state w of a CPM $\mathcal{M}$ if $\mathcal{M}, w \models \phi$. In this case we also say that ϕ *holds* in w if $\mathcal{M}$ is clear from the context. A formula ϕ is *true* in $\mathcal{M}$ if it is true in all states of $\mathcal{M}$. We denote by $\mathcal{M} \models \phi$. In this case we also say that ϕ *holds* in $\mathcal{M}$. A formula ϕ is *valid* in the class of conditional probability models (CPM) if for any CPM $\mathcal{M}$ it holds that $\mathcal{M} \models \phi$. In this case we write $\models \phi$.

An important fact about the universal modality is that a formula of the form $\Diamond\phi$ is satisfied in a state if and only if $[\![\phi]\!] \neq \varnothing$.

Theorem 3.4. For any state w of a model $\mathcal{M}$ the following conditions are equivalent:

1. $[\![\phi]\!] \neq \varnothing$.
2. $\mathcal{M}, w \models \Diamond\phi$.

An interesting fact about the operator L_1 is that it is a *normal modality*, i.e., (1) the formula schema $L_1(\phi \to \psi) \to (L_1\phi \to L_1\psi)$ is valid in any CPM; and (2) the rule 'If ϕ is valid in all CPM, then $L_1\phi$ is valid in all CPM' is true.

3.3 Axioms

System L_{cp} consists of the following axioms and rules:

($A0$) propositional calculus

($K_\Box$) $\Box(\phi \to \psi) \to (\Box\phi \to \Box\psi)$

($T_\Box$) $\Box\phi \to \phi$

($5_\Box$) $\Diamond\phi \to \Box\Diamond\phi$

($A1$) $L_0(\phi||\psi)$

($A2$) $L_r(\phi||\phi)$ for $0 \leq r \leq 1$

($A3$) $L_r(\phi \wedge \psi||\chi) \wedge L_t(\phi \wedge \neg\psi||\chi) \to L_{r+t}(\phi||\chi)$, for $r + t \leq 1$

($A4$) $\neg L_r(\phi \wedge \psi||\chi) \wedge \neg L_t(\phi \wedge \neg\psi||\chi) \to \neg L_{r+t}(\phi||\chi)$, for $r + t \leq 1$

($A5$) $\Diamond\psi \to (L_r(\phi||\psi) \to \neg L_s(\neg\phi||\psi))$, for $r + s > 1$

($C1$) $L_r(\phi||\psi \wedge \chi) \wedge L_s(\psi||\chi) \to L_{rs}(\phi \wedge \psi||\chi)$

($C2$) $\neg L_r(\phi||\psi \wedge \chi) \wedge \neg L_s(\psi||\chi) \to \neg L_{rs}(\phi \wedge \psi||\chi)$

($C_\Box$) $\Box(\phi \leftrightarrow \phi') \wedge \Box(\psi \leftrightarrow \psi') \to (L_r(\phi||\psi) \leftrightarrow L_r(\phi'||\psi'))$

($N_\Box$) If $\vdash \phi$, then $\vdash \Box\phi$

($ARCH$) If $\vdash \gamma \to \neg M_s(\phi||\psi)$ for all $s < r$, then $\vdash \gamma \to L_r(\phi||\psi)$.

($A0$) contains all tautologies and the rule of *modus ponens.*

The axioms ($A1$) to ($A5$) are essentially the axioms in the system presented in Zhou (2009). Let ψ be the *conditional part* of a formula of the form $L_r(\phi||\psi)$. Define the deduction rule (DIS) to be 'if $\vdash (\phi \leftrightarrow \psi)$ then $\vdash L_r\phi \leftrightarrow L_r\psi$'. Let the system L' consist of the axioms ($A1$) to ($A5$) with $\top$ as their conditional part (changing $\Diamond\psi$ for $\top$ in ($A5$)) and the inference rules (DIS) and ($ARCH$) with $\top$ as their conditional part. Then L' is the same system from Zhou (2009).

The restriction $\Diamond\phi$ in axiom ($A5$) is important in the proof of σ-additivity of the conditional probability function. For any A, B the following inequality holds $\mu(A|B) + \mu(X\setminus A|B) \geq \mu(A \cup (X \setminus A)|B)$. The other inequality ($\leq$) holds only when B is non-empty. Axiom ($A5$) is used for the proof of, e.g., Lemma 3.18 in the proof of completeness.

The axioms ($C1$) and ($C2$) were added to prove completeness in the conditional probability models and they are related to the two inequalities that define the conditional probability of an event.[1] The axioms ($K_\Box$) to ($5_\Box$) are the axioms of the standard $S5$

[1] $T(w, A \cap B|C) = T(w, A \mid B \cap C) \cdot T(w, B \mid C)$.

modal logic which is the correct and sound system for the universal modality (see van Benthem (2010)). The inference rule (DIS) from Zhou (2009) is derivable from axioms ($C_\Box$) and ($N_\Box$).

The inference rule ($ARCH$) holds because of the *Archimedean property* of the Real numbers, i.e., given any real number there is a rational number greater than the number given; or equivalently, between two real numbers there is a rational number.

A set Γ of formulas is *inconsistent* in L_{cp} if there are $\gamma_1, \ldots, \gamma_n \in \Gamma$ such that $\neg(\gamma_1 \wedge \ldots \wedge \gamma_n)$ is provable in L_{cp}, and Γ is *consistent* if it is not inconsistent. A single formula ϕ is (in)consistent if $\{\phi\}$ is (in)consistent.

Soundness

The proof of soundness consists in showing that each axiom of L_{cp} is true in $\mathcal{M}$ for any $\mathcal{M}$ in the class of conditional probability models and validity is preserved by the rules ($N_\Box$) and ($ARCH$).

Theorem 3.5. The system L_{cp} is sound.

Proof. We prove only the validity of axioms ($A3$) and ($C1$) since the others are routine verification.

Let $\mathcal{M} = \langle \Omega, \Sigma, T, v\rangle$ be a conditional probability model, and let $w \in \Omega$, $\phi, \psi \in \mathcal{L}$, $r \in [0,1] \cap \mathbb{Q}$:

$\cdot(A3) L_r(\phi \wedge \psi || \chi) \wedge L_t(\phi \wedge \neg\psi || \chi) \to L_{r+t}(\phi || \chi)$, for $r + t \leq 1$

Suppose that $\mathcal{M}, w \vDash \Diamond\chi$, $\mathcal{M}, w \vDash L_r(\phi \wedge \psi || \chi)$ and $\mathcal{M}, w \vDash L_t(\phi \wedge \neg\psi || \chi)$. Then $T(w, [\![\phi \wedge \psi]\!], [\![\chi]\!]) \geq r$ and $T(w, [\![\phi \wedge \neg\psi]\!], [\![\chi]\!]) \geq t$. Since $[\![\phi \wedge \psi]\!] \cup [\![\phi \wedge \neg\psi]\!] = [\![\phi]\!], [\![\phi \wedge \psi]\!] \cap [\![\phi \wedge \neg\psi]\!] = \varnothing$ and $[\![\chi]\!] \neq \varnothing$, adding each side of the two inequalities we have:

$$T(w, [\![\phi]\!], [\![\chi]\!]) \geq r + t$$

Since $r + t \leq 1$ we have that $\mathcal{M}, w \vDash L_{r+t}(\phi || \chi)$.

Suppose that $\mathcal{M}, w \vDash \neg\Diamond\chi$. Then $[\![\chi]\!]$ is empty which implies that $\mathcal{M}, w \vDash L_{r+t}(\phi || \chi)$ always holds (because $T(w, A, \varnothing) = 1$ for every $A \in \Sigma$.)

$\cdot(C1) L_r(\phi || \psi \wedge \chi) \wedge L_s(\psi || \chi) \to L_{rs}(\phi \wedge \psi || \chi)$

Suppose $\mathcal{M}, w \vDash L_r(\phi || \psi \wedge \chi)$ and $\mathcal{M}, w \vDash L_s(\psi || \chi)$, which is equivalent to $T(w, [\![\phi]\!], [\![\psi \wedge \chi]\!]) \geq r$ and $T(w, [\![\psi]\!], [\![\chi]\!]) \geq s$. Multiplying both inequalities we have $T(w, [\![\phi \wedge \psi]\!], [\![\chi]\!]) = T(w, [\![\phi]\!], [\![\psi \wedge \chi]\!]) \cdot T(w, [\![\psi]\!], [\![\chi]\!]) \geq rs$. The equality holds because $\mathcal{M}$ is a conditional probability space. The inequality implies that $\mathcal{M}, w \vDash L_{rs}(\phi \wedge \psi || \chi)$. $\Box$

In order to prove that the axiom ($A3$) is valid we analyzed two cases: $\mathcal{M} \models \Diamond\chi$ and $\mathcal{M} \models \neg\Diamond\chi$. This is equivalent to the cases $[\![\chi]\!] \neq \varnothing$ and $[\![\chi]\!] = \varnothing$.

The following theorems in this paragraph are technical results that we make use of a number of times for the proof of completeness.

Theorem 3.6. The following are provable in L_{cp}:

(i) $\Box(\phi \to \psi) \to (L_r(\phi||\chi) \to L_r(\psi||\chi))$.

(ii) $\Box(\phi \to \psi) \to (\neg L_r(\psi||\chi) \to \neg L_r(\phi||\chi))$.

(iii) $\Box\neg(\phi \wedge \psi) \to (L_r(\psi||\chi) \wedge L_s(\phi||\chi) \to L_{r+s}(\phi \wedge \psi||\chi))$, for $r + s \leq 1$.

(iv) $\neg L_r(\phi||\chi) \wedge \neg L_s(\psi||\chi) \to \neg L_{r+s}(\phi \vee \psi||\chi)$, for $r + s \leq 1$.

Proof. (i) $\Box(\phi \to \psi) \to (L_r(\phi||\chi) \to L_r(\psi||\chi))$:

1.	$(\phi \to \psi) \to (\phi \wedge \psi \leftrightarrow \phi)$	($A0$)
2.	$\Box(\phi \to \psi) \to \Box(\phi \wedge \psi \leftrightarrow \phi)$	1, ($N_\Box$)
3.	$\Box(\phi \wedge \psi \leftrightarrow \phi) \to (\phi \wedge \psi \leftrightarrow \phi)$	$T_\Box$
4.	$\Box(\phi \to \psi) \to (\phi \wedge \psi \leftrightarrow \phi)$	3,2,($A0$)
5.	$\Box(\phi \to \psi) \to (L_r(\phi \wedge \psi||\chi) \leftrightarrow L_r(\phi||\chi))$	4,($C_\Box$), ($N_\Box$)
6.	$L_r(\phi \wedge \psi||\chi) \wedge L_0(\neg\phi \wedge \psi||\chi) \to L_r(\psi||\chi)$	($A3$)
7.	$L_0(\neg\phi \wedge \psi||\chi)$	($A1$)
8.	$L_r(\phi \wedge \psi||\chi) \to L_r(\psi||\chi)$	6, 7, ($A0$)
9.	$\Box(\phi \to \psi) \to (L_r(\phi||\chi) \to L_r(\psi||\chi))$	5,8,($A0$)

(ii) Consequence of item (i).

(iii) $\Box\neg(\phi \wedge \psi) \to (L_r(\psi||\chi) \wedge L_s(\phi||\chi) \to L_{r+s}(\phi \wedge \psi||\chi))$:

1.	$\neg(\phi \wedge \psi) \to ((\phi \vee \psi) \wedge \psi \to (\phi \vee \psi) \wedge \neg\phi)$	($A0$)
2.	$\Box\neg(\phi \wedge \psi) \to \Box((\phi \vee \psi) \wedge \psi \to (\phi \vee \psi) \wedge \neg\phi)$	1,($N_\Box$)
3.	$\Box((\phi \vee \psi) \wedge \psi \to (\phi \vee \psi) \wedge \neg\phi) \to$ $(L_r((\phi \vee \psi) \wedge \psi||\chi) \to L_r((\phi \vee \psi) \wedge \neg\phi||\chi))$	(item (i))
4.	$\Box\neg(\phi \wedge \psi) \to (L_r((\phi \vee \psi) \wedge \psi||\chi) \to L_r((\phi \vee \psi) \wedge \neg\phi||\chi))$	2,3,($A0$)
5.	$L_r((\phi \vee \psi) \wedge \neg\phi||\chi) \wedge L_s((\phi \vee \psi) \wedge \phi||\chi) \to L_{r+s}(\phi \wedge \psi||\chi)$	($A3$)
6.	$\Box\neg(\phi \wedge \psi) \to$ $(L_r((\phi \vee \psi) \wedge \psi||\chi) \wedge L_s((\phi \vee \psi) \wedge \phi||\chi) \to L_{r+s}(\phi \wedge \psi||\chi))$	4,5,($A0$)
7.	$(\phi \vee \psi) \wedge \phi \leftrightarrow \phi$	($A0$)
8.	$(\phi \vee \psi) \wedge \psi \leftrightarrow \psi$	($A0$)
9.	$L_r((\phi \vee \psi) \wedge \psi||\chi) \leftrightarrow L_r(\phi||\chi)$	8,($N_\Box$),($C_\Box$)
10.	$L_s((\phi \vee \psi) \wedge \phi||\chi) \leftrightarrow L_s(\phi||\chi)$	7,($N_\Box$),($C_\Box$)
11.	$\Box\neg(\phi \wedge \psi) \to (L_r(\psi||\chi) \wedge L_s(\phi||\chi) \to L_{r+s}(\phi \wedge \psi||\chi))$	9,10,6,($A0$)

(iv) $\neg L_r(\phi||\chi) \wedge \neg L_s(\psi||\chi) \to \neg L_{r+s}(\phi \vee \psi||\chi)$, for $r + s \leq 1$:

1.	$((\phi \vee \psi) \wedge \neg\psi) \to \phi$	$(A0)$
2.	$((\phi \vee \psi) \wedge \psi) \to \psi$	$(A0)$
3.	$\neg L_r(\phi\|\|\chi) \to \neg L_r((\phi \vee \psi) \wedge \neg\psi\|\|\chi)$	(item (ii)),1
4.	$\neg L_t(\psi\|\|\chi) \to \neg L_t((\phi \vee \psi) \wedge \psi\|\|\chi)$	(item (ii)),2
5.	$\neg L_r((\phi \vee \psi) \wedge \neg\psi\|\|\chi) \wedge \neg L_t((\phi \vee \psi) \wedge \psi\|\|\chi) \to \neg L_{r+t}(\phi \vee \psi\|\|\chi)$	$(A4)$
6.	$\neg L_r(\phi\|\|\chi) \wedge \neg L_t(\psi\|\|\chi) \to \neg L_{r+t}(\phi \vee \psi\|\|\chi)$	3,4,$(A0)$

□

The following theorem presents some interesting and useful formulas that are theorems of L_{cp}.

Theorem 3.7. The following are provable in L_{cp}.

(i) $\Box(\psi \to \phi) \to L_r(\phi||\psi)$;

(ii) $\Box\phi \to L_1\phi$.

(iii) $L_r(\phi||\bot)$.

Proof. (i) $\Box(\psi \to \phi) \to L_r(\phi||\psi)$:

1.	$\Box(\phi \to \psi) \to (L_r(\phi\|\|\phi) \to L_r(\psi\|\|\phi))$	(Th.3.6-item (i))
2.	$L_r(\phi\|\|\phi)$	$(A2)$
3.	$\Box(\psi \to \phi) \to L_r(\phi\|\|\psi)$	1,2,$(A0)$

(ii) $\Box\phi \to L_1\phi$.

1.	$\phi \to (\phi \leftrightarrow \top)$	$(A0)$
2.	$\Box(\phi \to (\phi \leftrightarrow \top))$	$(N_\Box)$, 1
3.	$\Box\phi \to \Box(\phi \leftrightarrow \top)$	$(K_\Box), 2, (A0)$
4.	$\Box(\top \leftrightarrow \top)$	$(A0), (N_\Box)$
5.	$\Box(\phi \leftrightarrow \top) \to (L_1\phi \leftrightarrow L_1\top)$	$(C_\Box), 4, (A0)$
6.	$\Box\phi \to (L_1\phi \leftrightarrow L_1\top)$	$3, 5, (A0)$
7.	$L_1\top$	(A2)
8.	$\Box\phi \to L_1\phi$	$6, 7, (A0)$

(iii) $L_r(\phi||\bot)$.

1.	$\Box(\bot \to \phi) \to L_r(\phi\|\|\bot)$	(item (i))
2.	$\Box(\bot \to \phi)$	$(A0)$,$(N_\Box)$
3.	$L_r(\phi\|\|\bot)$	1,2,$(A0)$

□

Finally, we state two other theorems which we use for the proof of completeness in the next section.

Theorem 3.8. The following are provable in L_{cp}.
(i) $\neg L_r(\phi||\psi) \to M_r(\phi||\psi)$.
(ii) $L_r(\phi||\psi) \to L_s(\phi||\psi)$ if $r \geq s$.

Proof. (i) $\neg L_r(\phi||\psi) \to M_r(\phi||\psi)$:

1.	$\neg L_r(\top \wedge \phi||\psi) \wedge \neg L_{1-r}(\top \wedge \neg\phi||\psi) \to \neg L_1(\top||\psi)$	$(A4)$
2.	$\neg L_r(\phi||\psi) \wedge \neg L_{1-r}(\neg\phi||\psi) \to \neg L_1(\top||\psi)$	$(C_\Box)$, $(A0)$
3.	$L_1(\top||\psi)$	(Th.3.7-item (i)),$(K_\Box)$, $(A0)$
4.	$\neg L_r(\phi||\psi) \to M_r(\phi||\psi)$	2,3,$(A0)$

(ii) $L_r(\phi||\psi) \to L_s(\phi||\psi)$ if $r \geq s$:

If $r = s$, there is nothing to do. Suppose $r > s$,

1.	$\neg L_s(\phi \wedge \phi||\psi) \wedge \neg L_{r-s}(\phi \wedge \neg\phi||\psi) \to \neg L_r(\phi||\psi)$	$(A4)$
2.	$\Diamond\psi \to (L_r(\top||\psi) \to \neg L_{r-s}(\bot||\psi))$	$(A5)$
3.	$\Diamond\psi \to (L_r(\phi||\psi) \to L_s(\phi||\psi))$	2,(Th.3.7-item (i)),1,$(A0)$
4.	$\Box(\psi \to \bot) \to (L_r(\phi||\bot) \leftrightarrow L_r(\phi||\psi))$	$(C_\Box)$, $(A0)$
5.	$\Box\neg\psi \to L_r(\phi||\psi)$	4, (Th.3.7-item (iii)), $(A0)$
6.	$\Box\neg\psi \to L_s(\phi||\psi)$	$(C_\Box)$,(Th.3.7-item (iii)), $(A0)$
7.	$\Box\neg\psi \to (L_r(\phi||\psi) \to L_s(\phi||\psi))$	5,6,$(A0)$
8.	$(\Box\neg\psi) \vee (\Diamond\psi) \to (L_r(\phi||\psi) \to L_s(\phi||\psi))$	3,7,$(A0)$
9.	$L_r(\phi||\psi) \to L_s(\phi||\psi)$	8,$(A0)$

□

Completeness

The strategy for the proof of completeness is to build a canonical model (the states are the maximal consistent sets of formulas). Before the proof itself we sketch below some parts of it to help the reader. The proof is similar in many ways to the proofs in Heifetz and Mongin (2001), Zhou (2009) and Zhou (2014).

Completeness of Conditional Probability Logic: Proof Sketch

- Fix a consistent formula χ in $\mathcal{L}$. The goal is to construct a conditional probability model that satisfies χ.
- Define a finite language (finite up to provable equivalences) which is given in function of the propositional variables in χ and its highest number of nested L_r operators, the depth of χ. Call this finite language $\mathcal{L}(\chi)$.

In Zhou (2007) the knowledge operator K is added to probability logic and the finite language is defined in function of the number of nested operators L_r and K. Our proof uses the fact that any formula of $S5$ is equivalent to a formula with the highest number of nested $\Box$ less than or equal to one.

- Let Ω be the set of all maximum consistent set of formulas of $\mathcal{L}(\chi)$ and define an equivalence relation $\sim$ over Ω (this relation is the semantic counterpart of the universal modality $\Box$). Let Γ_χ be a maximal consistent set of formulas of $\mathcal{L}(\chi)$ containing χ. There are are possible choices, fix any.
- Let the canonical model be $\mathcal{M}_\chi = \langle \Omega_\chi, \Sigma_\chi, T_\chi \rangle$, where Ω_χ is the set of all maximal consistent set of formulas that are in the equivalence class of Γ_χ (i.e., $\Omega_\chi = \{\Delta \in \Omega : \Delta \sim \Gamma_\chi\}$), and Σ_χ is the set of elements of the form $[\phi]$; ϕ is a formula of the finite language and $[\phi] = \{\Delta \in \Omega_\chi : \phi \in \Delta\}$.

  Different from Zhou (2007) where the knowledge operator K is added to probability logic, we define the support of the model (the set Ω_χ) as one of the equivalence classes in the whole universe Ω, otherwise the universal modality cannot reach every state in the model.
- The definition of the conditional Markov kernel T_χ is a little bit more laborious. For each $\Gamma_i \in \Omega$ let Γ_i^∞ be a maximal consistent extension of Γ_i in the language $\mathcal{L}$ (Γ_i^∞ is a set of formulas of $\mathcal{L}$). Define $T_\chi(\Gamma_i, [\phi], [\psi]) = \alpha_{\phi,\psi}^{\Gamma_i^\infty}$ where

$$\alpha_{\phi,\psi}^{\Gamma_i^\infty} = \sup\{\alpha : L_\alpha(\phi||\psi) \in \Gamma_i^\infty\}.$$

  We spend most of the proof showing that as it is defined T_χ is indeed a conditional probability measure. We rely on some techniques from Zhou (2009), specially when referring to $T_\chi(\Gamma, [\phi], [\psi])$ with $[\psi] \neq \varnothing$.

  To bound the size of the set of states to a finite number we allow only formulas up to a fixed depth. The depth of a formula is the maximum of the number of L operators nested inside one another.

We define depth formally as follows.

Definition 3.9 (Local Language)**.** The *depth* $dp(\phi)$ of a formula ϕ is defined inductively:

- $dp(p) = 0$, for propositional letters p;
- $dp(\neg\phi) = dp(\phi)$;

- $dp(\phi_1 \wedge \phi_2) = \max\{dp(\phi_1), dp(\phi_2)\}$;
- $dp(\Box\phi) = dp(\phi)$;
- $dp(L_r(\phi||\psi)) = 1 + \max\{dp(\phi), dp(\psi)\}$.

Let the *index* of a formula of the form $L_r(\phi||\psi)$ be the rational number r. Remember that P is the set of propositional variables. Let $Q \subset P, q$ a positive rational number ($q > 0$) and d a natural number($d \geq 0$). We define a *local language* $\mathcal{L}(Q, q, d)$ to be the set of all formulas ϕ satisfying the following:

- Every propositional variable occurring in ϕ is in Q;
- Every index in ϕ is a multiple of $\frac{1}{q}$; and
- $dp(\phi) \leq d$.

The integer q is called the *accuracy* of the language $\mathcal{L}(Q, q, d)$.

For $\phi \in \mathcal{L}(Q, q, d)$, we write $\bar{\phi}$ for the equivalence class

$$\{\, \psi \in \mathcal{L}(Q, q, d) \mid \psi \leftrightarrow \phi \text{ is a theorem of } L_{cp} \,\}.$$

Lemma 3.10. If Q is finite, then $\{\, \bar{\phi} \mid \phi \in \mathcal{L}(Q, q, d) \,\}$ is finite.

Proof. This is proved by induction on d. Suppose $|Q| = n$. Note that formulas of $\mathcal{L}(Q, q, 0)$ are simply formulas of propositional modal logic over n propositional variables. By the *modal conjunctive normal form theorem* of Hughes and Cresswell (1996) we know that every modal formula is provably equivalent in S5 to a finite conjunction of formulas of the form

$$\beta \vee \Box\gamma_1 \vee \cdots \vee \Box\gamma_k \vee \Diamond\delta, \tag{†}$$

where $\beta, \gamma_1, \ldots, \gamma_k, \delta$ are propositional formulas. Over n propositional variables, there are at most

$$f(n) = 2^{2^n} \cdot 2^{2^{2^n}} \cdot 2^{2^n}$$

non-S5-equivalent disjunctions of the form (†), and at most

$$g(n) = 2^{f(n)}$$

non-S5-equivalent finite conjunctions of such disjunctions. Since L_{cp} contains S5, this takes care of the induction basis. Let $h(n, q, 0) = g(n)$.

Now assume that every formula in $\mathcal{L}(Q, q, d)$ is provably equivalent in L_{cp} to one of a finite number $h(n, q, d)$ of formulas. Call a formula in $\mathcal{L}(Q, q, d + 1)$ a *modal atom* if it is a propositional variable in Q or a formula of the form $L_{i/q}(\psi||\chi)$, where $0 \leq i \leq q$ and $\psi, \chi \in \mathcal{L}(Q, q, d)$. Evidently, every formula ϕ in $\mathcal{L}(Q, q, d+1)$ is constructed from modal atoms using $\Box$, $\neg$, and $\wedge$ only. By the modal conjunctive normal form theorem, ϕ is provably equivalent to a finite conjunction of formulas of the form (†), where $\beta, \gamma_1, \ldots, \gamma_k, \delta$ are Boolean combinations of modal atoms. Since there are at most $(q+1) \cdot h(n, q, d)^2$ non-equivalent formulas of the form $L_{i/q}(\psi||\chi)$ with $\psi, \chi \in \mathcal{L}(Q, q, d)$, the number of non-equivalent modal atoms in $\mathcal{L}(Q, q, d)$ is at most $n + (q+1) \cdot h(n, q, d)^2$. So the number of non-equivalent formulas in $\mathcal{L}(Q, q, d+1)$ is bounded by

$$h(n, q, d+1) = g(n + (q+1) \cdot h(n, q, d)^2).$$

□

By Lemma 3.10, if $\Gamma \subseteq \mathcal{L}(Q, q, d)$ for some finite Q, there is a finite subset $\{\psi_1, \ldots, \psi_k\}$ of Γ such that every formula in Γ is provably equivalent to one of $\psi_1, \ldots, \psi_k$. We denote by $\bigwedge \Gamma$ the conjunction $\psi_1 \wedge \cdots \wedge \psi_k$. Note that $\bigwedge \Gamma \in \mathcal{L}(Q, q, d)$, and up to provable equivalence, $\bigwedge \Gamma$ is independent of the choice of $\psi_1, \ldots, \psi_k$.

A set $\Gamma \subseteq \mathcal{L}(Q, q, d)$ is said to be a *maximal consistent* subset of $\mathcal{L}(Q, q, d)$ if Γ is consistent and there exists no consistent Γ' such that $\Gamma \subsetneq \Gamma' \subset \mathcal{L}(Q, q, d)$. If Γ is a maximal consistent subset of $\mathcal{L}(Q, q, d)$, then for every $\Delta \subset \Gamma$, $\bigwedge \Delta$ belongs to Γ.

Lemma 3.11. Let Γ, Γ' be maximal consistent subsets of $\mathcal{L}(Q, q, d)$. If $\bigwedge \Gamma \in \Gamma'$, then $\Gamma = \Gamma'$.

Proof. If $\bigwedge \Gamma \in \Gamma'$, then $\Gamma \subset \Gamma'$. Since Γ is maximal consistent and Γ' is consistent, $\Gamma = \Gamma'$. □

If $\chi \in \mathcal{L}$, then $\mathcal{L}(\chi)$ is defined as $\mathcal{L}(P_\chi, q_\chi, d_\chi)$, where P_χ is the set of all propositional letters in ψ, the index q_χ is the least common multiple of all the denominators of the indices in χ and d_χ is the depth of χ .

Fix χ in $\mathcal{L}$ and suppose it is consistent in L_{cp}. Let Ω be the set of all maximal consistent sets of formulas in $\mathcal{L}(\chi)$.

Given a maximal consistent set of formulas Γ in Ω, define $U(\Gamma) = \{\phi \in \Gamma : \phi$ is of the form $\neg\Box\psi$ or $\Box\psi\}$ and for any $\Gamma, \Delta \in \Omega$ we say that $\Gamma \sim \Delta$ iff $U(\Gamma) = U(\Delta)$. Let Γ_χ be a maximal consistent set of formulas that contains χ. Note that the choice of Γ_χ is not unique. Define $\Omega_\chi = \{\Delta \in \Omega : \Delta \sim \Gamma_\chi\}$. Let $[\phi]_\chi = \{\Delta \in \Omega_\chi : \phi \in \Delta\}$ and $\Sigma_\chi = \{[\phi]_\chi : \phi \in \mathcal{L}(\chi)\}$ ($\Sigma_\chi^* = \Sigma_\chi \setminus \{\varnothing\}$). Denote by $\gamma_\Box$ the conjunction $\bigwedge U(\Gamma_\chi)$.

Lemma 3.12. $\Omega_\chi = \{\Delta \in \Omega : \gamma_\Box \in \Delta\} (\doteq [\gamma_\Box])$.

Proof. Let $\Delta \in \Omega_\chi$. Then $U(\Delta) = U(\Gamma_\chi)$, and clearly $\bigwedge U(\Delta) \in \Delta$, hence $\gamma_\Box \in \Delta$, i.e., $\Delta \in [\gamma_\Box]$.

For the other inclusion let $\Delta \in [\gamma_\Box]$ and let us show that $U(\Delta) = U(\Gamma_\chi)$. Consider $\phi \in U(\Gamma_\chi)$. Clearly, $\vdash \gamma_\Box \to \phi$, so $\phi \in \Delta$. Since ϕ is of the form $\Box\psi$, we have $\phi \in U(\Delta)$. Now let $\phi \in U(\Delta)$ and suppose it is not in $U(\Gamma_\chi)$. By maximality, $\Gamma_\chi \vdash \neg\phi$. Then $\neg\phi \in U(\Gamma_\chi)$ and $\vdash \gamma_\Box \to \neg\phi$. It follows that $\neg\phi \in \Delta$, which is a contradiction. □

Lemma 3.13. $\Sigma_\chi = \mathcal{P}(\Omega_\chi)$.

Proof. It is easy to see that if $\phi \in \mathcal{L}(\chi)$, then $[\phi]_\chi \in \mathcal{P}(\Omega_\chi)$.

Let $X = \{\Gamma_1, \ldots, \Gamma_n\} \subseteq \Omega_\chi$. Let γ_i be $\bigwedge \Gamma_i$ and let ϕ be $\bigvee_{i=1}^n \gamma_i$. Clearly, $\phi \in \Gamma_i$ for all $i = 1, \ldots, n$. If $\phi \in \Delta$ for some $\Delta \in \Omega_\chi$, then some $\gamma_i \in \Delta$. By Lemma 3.11, $\Gamma_i = \Delta$. It follows that $X = [\phi]_\chi$. □

The pair $(\Omega_\chi, \Sigma_\chi)$ is the measurable space for our canonical model. For the conditional Markov kernel we need some steps that we prove first.

Let Δ be a maximal consistent set of formulas in $\mathcal{L}(Q, q, d+1)$ and let ϕ and ψ be formulas in $\mathcal{L}(Q, q, d)$. Define:

$$\alpha^\Delta_{\phi,\psi} = \max\{\alpha : L_\alpha(\phi||\psi) \in \Delta\},$$

$$\beta^\Delta_{\phi,\psi} = \min\{\beta : M_\beta(\phi||\psi) \in \Delta\}.$$

Let $\mathcal{L}(\chi^+)$ be the language $\mathcal{L}(P_\chi, q_\chi, d_\chi + 1)$. Note that $\mathcal{L}(\chi) \subset \mathcal{L}(\chi^+) \subset \mathcal{L}$. For each Γ in Ω let Γ^+ be a maximal consistent extension of Γ in $\mathcal{L}(\chi^+)$. Note that this choice is not unique; fix one extension for each $\Gamma \in \Omega$. Note that $L_{\alpha^{\Gamma^+}_{\phi,\psi}}(\phi||\psi) \in \Gamma^+$ and $M_{\beta^{\Gamma^+}_{\phi,\psi}}(\phi||\psi) \in \Gamma^+$.

Item (a.) in the next Lemma is essentially Lemma 3.11 from Zhou (2009).

Lemma 3.14. For any $\Gamma \in \Omega_\chi$ and $\phi, \psi \in \mathcal{L}(\chi)$:

(a.) If $\Diamond\psi \in \Gamma$, then either $\beta^{\Gamma^+}_{\phi,\psi} = \alpha^{\Gamma^+}_{\phi,\psi}$ or $\beta^{\Gamma^+}_{\phi,\psi} = \alpha^{\Gamma^+}_{\phi,\psi} + \frac{1}{q_\chi}$;

(b.) If $\Diamond\psi \notin \Gamma$, then $\alpha^{\Gamma^+}_{\phi,\psi} = 1$ and $\beta^{\Gamma^+}_{\phi,\psi} = 0$.

Proof. (a.)

($\alpha^{\Gamma^+}_{\phi,\psi} \le \beta^{\Gamma^+}_{\phi,\psi}$) Suppose $\alpha^{\Gamma^+}_{\phi,\psi} > \beta^{\Gamma^+}_{\phi,\psi}$. It follows that $(1 - \beta^{\Gamma^+}_{\phi,\psi}) + \alpha^{\Gamma^+}_{\phi,\psi} > 1$.

We have that $L_{\alpha^{\Gamma^+}_{\phi,\psi}}(\phi||\psi) \in \Gamma^+$ and by (A5), $\neg L_{1-\beta^{\Gamma^+}_{\phi,\psi}}(\neg\phi||\psi) = \neg M_{\beta^{\Gamma^+}_{\phi,\psi}}(\phi||\psi) \in \Gamma^+$, but this contradicts the fact that $M_{\beta^{\Gamma^+}_{\phi,\psi}}(\phi||\psi) \in \Gamma^+$.

($\alpha^{\Gamma^+}_{\phi,\psi} \geq \beta^{\Gamma^+}_{\phi,\psi} - \frac{1}{q_\chi}$) Let $s = \alpha^{\Gamma^+}_{\phi,\psi} + \frac{1}{q_\chi}$. Then s is a multiple of $\frac{1}{q_\chi}$ and $\alpha^{\Gamma^+}_{\phi,\psi} < s$, so $L_s(\phi||\psi) \notin \Gamma^+$. Since Γ^+ is a maximal consistent subset of $\mathcal{L}(P_\chi, q_\chi, d+1)$, we must have $\neg L_s(\phi||\psi) \in \Gamma^+$. By Theorem 3.8, $M_s(\phi||\psi) \in \Gamma^+$, so $s \geq \beta^{\Gamma^+}_{\phi,\psi}$, i.e., $\alpha^{\Gamma^+}_{\phi,\psi} \geq \beta^{\Gamma^+}_{\phi,\psi} - \frac{1}{q_\chi}$.

(b.)

If $\Diamond\psi \notin \Gamma$, then $\Box\neg\psi \in \Gamma$. Since $\neg\psi \to (\psi \leftrightarrow \bot)$ is a tautology, $\Box(\psi \leftrightarrow \bot) \in \Gamma$ by $(N_\Box)$ and $(K_\Box)$. Since for any γ, $L_1(\gamma||\bot)$ is a theorem of L_{cp}, we get $L_1(\phi||\psi) \in \Gamma$ and $M_0(\phi||\psi) \in \Gamma$ by $(C_\Box)$. So $\alpha^{\Gamma^+}_{\phi,\psi} = 1$ and $\beta^{\Gamma^+}_{\phi,\psi} = 0$. □

Definition 3.15. Let Γ^∞ be a maximal consistent extension of Γ^+ in $\mathcal{L}$ and ϕ, ψ be formulas in $\mathcal{L}(\chi)$. Define

$$\alpha^{\Gamma^\infty}_{\phi,\psi} = \sup\{\alpha : L_\alpha(\phi||\psi) \in \Gamma^\infty\},$$
$$\beta^{\Gamma^\infty}_{\phi,\psi} = \inf\{\beta : M_\beta(\phi||\psi) \in \Gamma^\infty\}.$$

Lemma 3.16. If r is a rational number and $r < \alpha^{\Gamma^\infty}_{\phi,\psi}$, then $L_r(\phi||\psi) \in \Gamma^\infty$.

Proof. Let s be a rational number such that $r \leq s < \alpha^{\Gamma^\infty}_{\phi,\psi}$ and $L_s(\phi||\psi) \in \Gamma^\infty$. Note that if $L_s(\phi||\psi) \notin \Gamma^\infty$ for all such s, then we can derive a contradiction with the fact that $\alpha^{\Gamma^\infty}_{\phi,\psi}$ is the supremum.

By Theorem 3.8.(ii), $L_s(\phi||\psi) \to L_r(\phi||\psi)$ is a theorem, hence also in Γ^∞. By the maximality of Γ^∞, it holds that $L_r(\phi||\psi) \in \Gamma^\infty$. □

The next Lemma is based on Lemma 3.12 from Zhou (2009).

Lemma 3.17. Let $\Diamond\psi \in \Gamma$ and $\phi, \psi \in \mathcal{L}(\chi)$. Then $\alpha^{\Gamma^\infty}_{\phi,\psi} = \beta^{\Gamma^\infty}_{\phi,\psi}$

Proof. Suppose $\alpha^{\Gamma^\infty}_{\phi,\psi} < \beta^{\Gamma^\infty}_{\phi,\psi}$. Then there is a rational r such that $\alpha^{\Gamma^\infty}_{\phi,\psi} < r < \beta^{\Gamma^\infty}_{\phi,\psi}$, which implies $L_r(\phi||\psi) \notin \Gamma^\infty$, i.e., $\neg L_r(\phi||\psi) \in \Gamma^\infty$. Hence $M_r(\phi||\psi) \in \Gamma^\infty$, contradicting $r < \beta^{\Gamma^\infty}_{\phi,\psi}$.

If $\alpha^{\Gamma^\infty}_{\phi,\psi} > \beta^{\Gamma^\infty}_{\phi,\psi}$, let r_1 and r_2 be rationals such that $\alpha^{\Gamma^\infty}_{\phi,\psi} > r_1 > r_2 > \beta^{\Gamma^\infty}_{\phi,\psi}$. Then $L_{r_1}(\phi||\psi) \in \Gamma^\infty$ and by $(A5)$[1] we have $\neg M_{r_2}(\phi||\psi) \in \Gamma^\infty$, contradicting $r_2 > \beta^{\Gamma^\infty}_{\phi,\psi}$. □

The next Lemma is similar to Lemma 3.15 from Zhou (2009); the difference is the added hypothesis $\Diamond\psi \in \Gamma$.

Lemma 3.18. Let $\phi_1, \phi_2, \psi \in \mathcal{L}(\chi)$ and $\Gamma \in \Omega_\chi$. If $\neg\Diamond(\phi_1 \wedge \phi_2) \in \Gamma$ and $\Diamond\psi \in \Gamma$, then

$$\alpha^{\Gamma^\infty}_{\phi_1\vee\phi_2,\psi} = \alpha^{\Gamma^\infty}_{\phi_1,\psi} + \alpha^{\Gamma^\infty}_{\phi_2,\psi}. \tag{3.1}$$

[1]Since $r_1 > r_2$ implies $(1 - r_2) + r_1 > 1$, an instance of $(A5)$ is $L_{r_1}(\phi||\psi) \to \neg L_{1-r_2}(\neg\phi||\psi)$.

Proof. Let α_1, α_2 and α_+ denote $\alpha^{\Gamma^\infty}_{\phi_1,\psi}, \alpha^{\Gamma^\infty}_{\phi_2,\psi}$ and $\alpha^{\Gamma^\infty}_{\phi_1\vee\phi_2,\psi}$, respectively. So we only need to show that $\alpha_1 + \alpha_2 = \alpha_+$.

Suppose $\alpha_1 + \alpha_2 < \alpha_+$. Then there are rationals $\alpha'_1 > \alpha_1$ and $\alpha'_2 > \alpha_2$ such that $\alpha'_1 + \alpha'_2 < \alpha_+$. It follows that $L_{\alpha'_1}(\phi_1||\psi) \notin \Gamma^\infty$ and hence $\neg L_{\alpha'_1}(\phi_1||\psi) \in \Gamma^\infty$. Similarly $\neg L_{\alpha'_2}(\phi||\psi) \in \Gamma^\infty$; by Theorem 3.6 it holds that $\neg L_{\alpha'_1+\alpha'_2}(\phi_1 \vee \phi_2||\psi) \in \Gamma^\infty$. That is a contradiction because $\alpha_1 + \alpha_2 < \alpha_+$ and α_+ is the greatest lower bound of $\{r : L_r(\phi_1 \vee \phi_2||\psi) \in \Gamma^\infty\}$.

Suppose that $\alpha_1 + \alpha_2 > \alpha_+$. Then there are rationals $\alpha''_1 < \alpha_1$ and $\alpha''_2 < \alpha_2$ such that $\alpha''_1 + \alpha''_2 > \alpha_+$. It follows that $L_{\alpha''_1}(\phi_1||\psi), L_{\alpha''_2}(\phi_2||\psi) \in \Gamma^\infty$. We want to prove that $\alpha''_1 + \alpha''_2 \leq 1$. Suppose that $\alpha''_1 + \alpha''_2 > 1$; since $\Diamond(\phi_1 \to \neg\phi_2) \in \Gamma$, by Theorem 3.6 we have $L_{\alpha''_1}(\phi_1||\psi) \to L_{\alpha''_1}(\neg\phi_2||\psi) \in \Gamma$ and clearly $L_{\alpha''_1}(\neg\phi_2||\psi) \in \Gamma^\infty$. By axiom $(A5)$,

$$\vdash \Diamond\psi \to (L_{\alpha''_1}(\neg\phi_2||\psi) \to \neg L_{\alpha''_2}(\phi_2||\psi));$$

we know that $\Diamond\psi, L_{\alpha''_1}(\phi_2||\psi) \in \Gamma^\infty$, hence we have $\neg L_{\alpha''_2}(\phi_2||\psi) \in \Gamma^\infty$ which is a contradiction. Therefore, we have $\alpha''_1 + \alpha''_2 \leq 1$. By Theorem 3.6, $L_{\alpha''_1+\alpha''_2}(\phi_1 \vee \phi_2||\psi) \in \Gamma^\infty$. But this is impossible because $\alpha''_1 + \alpha''_2 > \alpha_+$ and α_+ is the greatest lower bound of $\{r : L_r(\phi_1 \vee \phi_2||\psi) \in \Gamma^\infty\}$. □

Lemma 3.19. For any $\phi, \psi \in \mathcal{L}(\chi)$ and for any $\Gamma \in \Omega_\chi$ the following equation holds:

$$\alpha^{\Gamma^\infty}_{\phi_1\wedge\phi_2,\psi} = \alpha^{\Gamma^\infty}_{\phi_1,\phi_2\wedge\psi} \cdot \alpha^{\Gamma^\infty}_{\phi_2,\psi}$$

Proof. Suppose $\alpha^{\Gamma^\infty}_{\phi_1\wedge\phi_2,\psi} < \alpha^{\Gamma^\infty}_{\phi_1,\phi_2\wedge\psi} \cdot \alpha^{\Gamma^\infty}_{\phi_2,\psi}$.

Let x, y be rationals such that $x \leq \alpha^{\Gamma^\infty}_{\phi_1,\phi_2\wedge\psi}, y \leq \alpha^{\Gamma^\infty}_{\phi_2,\psi}$ and $\alpha^{\Gamma^\infty}_{\phi_1\wedge\phi_2,\psi} < xy$. Then $L_x(\phi_1||\phi_2 \wedge \psi) \in \Gamma^\infty$, $L_y(\phi_2||\psi) \in \Gamma^\infty$ and $L_{xy}(\phi_1 \wedge \phi_2||\psi) \notin \Gamma^\infty$. Since Γ^∞ is a maximal consistent subset of $\mathcal{L}$, by axiom $(C1)$ we have $L_{xy}(\phi_1 \wedge \phi_2||\psi) \in \Gamma^\infty$, a contradiction with the consistency of Γ^∞.

Suppose $\alpha^{\Gamma^\infty}_{\phi_1\wedge\phi_2,\psi} > \alpha^{\Gamma^\infty}_{\phi_1,\phi_2\wedge\psi} \cdot \alpha^{\Gamma^\infty}_{\phi_2,\psi}$.

Let x, y be rationals such that $x > \alpha^{\Gamma^\infty}_{\phi_1,\phi_2\wedge\psi}$, $y > \alpha^{\Gamma^\infty}_{\phi_2,\psi}$, and $\alpha^{\Gamma^\infty}_{\phi_1\wedge\phi_2,\psi} > xy$. Then $L_x(\phi_1||\phi_2 \wedge \psi) \notin \Gamma^\infty$, $L_y(\phi_2||\psi) \notin \Gamma^\infty$, and $L_{xy}(\phi_1 \wedge \phi_2||\psi) \in \Gamma^\infty$. Since Γ^∞ is a maximal consistent subset of $\mathcal{L}$, we have $\neg L_x(\phi_1||\phi_2 \wedge \psi) \in \Gamma^\infty$ and $\neg L_y(\phi_2||\psi) \in \Gamma^\infty$. By $(C2)$, we get $\neg L_{xy}(\phi_1 \wedge \phi_2||\psi) \in \Gamma^\infty$, contradicting the consistency of Γ^∞. □

For each $\Gamma \in \Omega_\chi$ and each $\phi, \psi \in \mathcal{L}(\chi)$ we want to define the conditional Markov kernel T by $T(\Gamma, [\phi]_\chi, [\psi]_\chi) = \alpha^{\Gamma^\infty}_{\phi,\psi}$ (Definition 3.22). Lemmas 3.21 and 3.24 below show that T as proposed makes sense.

Lemma 3.20. The following is a theorem in L_{cp}

$$\gamma_\Box \to \Box\gamma_\Box.$$

Proof. Recall that $\gamma_\Box$ is a finite conjunction of formulas of the form $\neg\Box\delta$ and $\Box\delta$. Since $\Box\delta \to \Box\Box\delta$, $\neg\Box\delta \to \Box\neg\Box\delta$ and $(\Box\delta_1 \wedge \Box\delta_2) \to \Box(\delta_1 \wedge \delta_2)$ are theorems of modal logic S5, $\gamma_\Box \to \Box\gamma_\Box$ is a theorem of L_{cp}. □

Lemma 3.21. For any $\Gamma \in \Omega_\chi$ and for any $\phi, \psi \in \chi$, $\alpha^{\Gamma^\infty}_{\phi\wedge\gamma_\Box,\psi\wedge\gamma_\Box} = \alpha^{\Gamma^\infty}_{\phi,\psi}$.

Proof. Since $\gamma_\Box \to (\phi \leftrightarrow (\phi \wedge \gamma_\Box))$ is an instance of a propositional tautology, we get $\Box\gamma_\Box \to \Box(\phi \leftrightarrow (\phi \wedge \gamma_\Box))$ by $(N_\Box)$ and $(K_\Box)$. For any $\Gamma \in \Omega_\chi$, by Lemma 3.20, $\Box\gamma_\Box \in \Gamma$. Hence it holds that $\Box(\phi \leftrightarrow (\phi \wedge \gamma_\Box)) \in \Gamma$.

Analogously $\Box(\psi \leftrightarrow (\psi \wedge \gamma_\Box)) \in \Gamma$.

By axiom $(C_\Box)$ we have $L_r(\phi||\psi) \leftrightarrow L_r(\phi \wedge \gamma_\Box||\psi \wedge \gamma_\Box) \in \Gamma^\infty$. Therefore $\alpha^{\Gamma^\infty}_{\phi,\psi} = \alpha^{\Gamma^\infty}_{\phi\wedge\gamma_\Box,\psi\wedge\gamma_\Box}$. □

Definition 3.22. Let $\Gamma \in \Omega_\chi$ and $\phi, \psi \in \mathcal{L}(\chi)$. Define $T_\chi(\Gamma, [\phi]_\chi, [\psi]_\chi) = \alpha^{\Gamma^\infty}_{\phi,\psi}$.

Note that the definition of T_χ depends on the choice of Γ^∞ for each Γ. We will later select a particular choice with a certain desirable property (Lemma 3.27), but we will first prove some facts about T_χ that do not depend on any particular choice.

The proof for the next lemma is straightforward.

Lemma 3.23. For any $\phi, \psi \in \mathcal{L}(\chi)$, $\vdash \phi \wedge \gamma_\Box \to \psi \wedge \gamma_\Box$ iff $[\phi]_\chi \subseteq [\psi]_\chi$.

The proof of the next Lemma is similar to Lemma 3.14 from Zhou (2009).

Lemma 3.24. T_χ as defined above is well-defined.

Proof. If $[\phi]_\chi = [\phi']_\chi$ and $[\psi]_\chi = [\psi']_\chi$, then by Lemma 3.23 we have $\vdash \phi \wedge \gamma_\Box \leftrightarrow \phi' \wedge \gamma_\Box$ and $\vdash \psi \wedge \gamma_\Box \leftrightarrow \phi' \wedge \gamma_\Box$. By $(N_\Box)$ and axiom $(C_\Box)$ we have $\vdash (L_r(\phi_1 \wedge \gamma_\Box||\psi \wedge \gamma_\Box) \leftrightarrow L_r(\phi' \wedge \gamma_\Box||\psi' \wedge \gamma_\Box)) \wedge (M_s(\phi \wedge \gamma_\Box||\psi \wedge \gamma_\Box) \leftrightarrow M_s(\phi' \wedge \gamma_\Box||\psi' \wedge \gamma_\Box))$, and hence $\alpha^{\Gamma^\infty}_{\phi\wedge\gamma_\Box,\psi\wedge\gamma_\Box} = \alpha^{\Gamma^\infty}_{\phi'\wedge\gamma_\Box,\psi'\wedge\gamma_\Box}$. By Lemma 3.21 we get $T_\chi(\Gamma, [\phi]_\chi, [\psi]_\chi) = T_\chi(\Gamma, [\phi']_\chi, [\psi']_\chi)$. □

To see that $T_\chi(\cdot, [\phi]_\chi, [\psi]_\chi)$ is a Σ_χ-measurable function it is enough to note that by Lemma 3.13 every subset of Ω_χ is measurable.

To show that for any Γ, $T_\chi(\Gamma, \cdot, \cdot)$ is a conditional probabilistic measure we first should note that the model is finite. Therefore, σ-additivity is equivalent to finite additivity.

To prove that $T_\chi(\Gamma)$ is additive we should prove that if $B \in \Sigma^*_\chi$ and $A_1, A_2 \in \Sigma_\chi$ are such that $A_1 \cap A_2 = \varnothing$, then $T_\chi(\Gamma, A_1 \cup A_2, B) = T_\chi(\Gamma, A_1, B) + T_\chi(\Gamma, A_2, B)$.

We finish proving that T_χ is a conditional Markov kernel with the next lemma.

Lemma 3.25. For any $\Gamma \in \Omega_\chi$:

(a.) If $A, B \in \Sigma_\chi$, then $T_\chi(\Gamma, A, B) \geq 0$ and $T_\chi(\Gamma, B, B) = 1$;

(b.) if $B \in \Sigma_\chi^*$ and $A_1, A_2 \in \Sigma_\chi$ are such that $A_1 \cap A_2 = \varnothing$, then $T_\chi(\Gamma, A_1 \cup A_2, B) = T_\chi(\Gamma, A_1, B) + T_\chi(\Gamma, A_2, B)$;

(c.) if $A, B, C \in \Sigma_\chi$, then

$$T_\chi(\Gamma, A \cap B, C) = T_\chi(\Gamma, A, B \cap C) \cdot T_\chi(\Gamma, B, C).$$

Proof. a. Since $L_1(\psi||\psi) \in \Gamma^+$, it is easy to see that $T_\chi(\Gamma, B, B) = 1$.

b. If $B \in \Sigma_\chi^*$, then there is a $\psi \in \mathcal{L}(\chi)$ such that $B = [\psi]_\chi$. Since $B \neq \varnothing$, there is a $\Delta \in \Omega_\chi$ such that $\psi \in \Delta$. Suppose that $\neg\Diamond\psi \in \Gamma$, then $\Box\neg\psi \in \Gamma$. Clearly $\Box\neg\psi \in U(\Gamma) = U(\Delta)$; and by axiom ($T_\Box$) we have $\neg\psi \in \Delta$, which is a contradiction. Therefore $\Diamond\psi \in \Gamma$.

If $\phi_1, \phi_2 \in \mathcal{L}(\chi)$ are such that $A_1 = [\phi_1]_\chi$ and $A_2 = [\phi_2]_\chi$, then we have that $\vdash \neg((\phi_1 \wedge \gamma_\Box) \wedge (\phi_2 \wedge \gamma_\Box))$. We have to prove that

$$T_\chi(\Gamma, [\phi_1 \wedge \gamma_\Box]_\chi \cup [\phi_2 \wedge \gamma_\Box]_\chi, [\psi]_\chi) = T_\chi(\Gamma, [\phi_1 \wedge \gamma_\Box]_\chi, [\psi]_\chi) + T_\chi(\Gamma, [\phi_2 \wedge \gamma_\Box]_\chi, [\psi]_\chi).$$

That is the same as $\alpha^{\Gamma\infty}_{\phi_1\wedge\gamma_\Box\vee\phi_2\wedge\gamma_\Box,\psi} = \alpha^{\Gamma\infty}_{\phi_1\wedge\gamma_\Box,\psi} + \alpha^{\Gamma\infty}_{\phi_2\wedge\gamma_\Box,\psi}$, which is true by Lemma 3.18. By Lemma 3.21 we have the desired equality:

$$\alpha^{\Gamma\infty}_{\phi_1\vee\phi_2,\psi} = \alpha^{\Gamma\infty}_{\phi_1,\psi} + \alpha^{\Gamma\infty}_{\phi_2,\psi}.$$

c. If $\phi_1, \phi_2, \psi \in \mathcal{L}(\chi)$ are such that $A = [\phi_1]_\chi$, $B = [\phi_2]_\chi$ and $C = [\psi]_\chi$, then we have to prove

$$T_\chi(\Gamma, [\phi_1 \wedge \phi_2]_\chi, [\psi]_\chi) = T_\chi(\Gamma, [\phi_1]_\chi, [\phi_2 \wedge \psi]_\chi) \cdot T_\chi(\Gamma, [\phi_2]_\chi, [\psi]_\chi).$$

It is enough to notice that the following equality holds by Lemma 3.19:

$$\alpha^{\Gamma\infty}_{\phi_1\wedge\phi_2,\psi} = \alpha^{\Gamma\infty}_{\phi_1,\phi_2\wedge\psi} \cdot \alpha^{\Gamma\infty}_{\phi_2,\psi}.$$

$\Box$

The next two Lemmas greatly simplify Lemma 3.16 from Zhou (2009). We explain with more details the differences after the proof.

Lemma 3.26. Let Γ be in Ω_χ and let $\Phi = \{\phi_1, ..., \phi_n\}$ be such that $\Delta = \Gamma^+ \cup \Phi$ is consistent, where each ϕ_i is a formula in $\mathcal{L}$. For any $\phi, \psi \in \mathcal{L}(\chi)$, if $\alpha^{\Gamma+}_{\phi,\psi} < \beta^{\Gamma+}_{\phi,\psi}$, then there is a rational $r < \beta^{\Gamma+}_{\phi,\psi}$ such that $\Delta \cup \{M_r(\phi||\psi)\}$ is consistent.

Proof. Let Δ be as in the hypothesis and suppose that ϕ and ψ are such that $\alpha^{\Gamma^+}_{\phi,\psi} < \beta^{\Gamma^+}_{\phi,\psi}$. Abbreviate $\bigwedge \Gamma^+ \wedge \phi_1 \wedge ... \wedge \phi_n$ by $\bigwedge \Delta$ the conjunction of all the formulas in Δ.

First remember that if $\alpha^{\Gamma^+}_{\phi,\psi} < \beta^{\Gamma^+}_{\phi,\psi}$, then $\neg L^{\Gamma^+}_{\beta_{\phi,\psi}}(\phi||\psi) \in \Gamma^+$. Hence

$$\vdash \bigwedge \Gamma^+ \to \neg L^{\Gamma^+}_{\beta_{\phi,\psi}}(\phi||\psi). \tag{†}$$

It follows that, since Δ is consistent $\nvdash \bigwedge \Delta \to L_{\beta^{\Gamma^+}_{\phi,\psi}}(\phi||\psi)$.

On the other hand suppose that for all $r < \beta^{\Gamma^+}_{\phi,\psi}$, $\vdash \bigwedge \Delta \to \neg M_r(\phi||\chi)$. Then by the (ARCH) rule we have $\vdash \bigwedge \Delta \to L_{\beta^{\Gamma^+}_{\phi,\psi}}(\phi||\psi)$, a contradiction with (†). Therefore there is an $r_0 < \beta^{\Gamma^+}_{\phi,\psi}$ such that $\Delta \cup \{M_{r_0}(\phi||\psi)\}$ is consistent. □

The following property is a modified version of property (E) from Zhou (2009).

Lemma 3.27. For each Γ in Ω_χ there is a maximal consistent Θ^∞ extension of Γ^+ in $\mathcal{L}$ such that the following property is satisfied:

(F): for any $\phi, \psi \in \mathcal{L}(\chi)$, if $\alpha^{\Gamma^+}_{\phi,\psi} < \beta^{\Gamma^+}_{\phi,\psi}$, then $\alpha^{\Gamma^+}_{\phi,\psi} \leq \alpha^{\Theta^\infty}_{\phi,\psi} = \beta^{\Theta^\infty}_{\phi,\psi} < \beta^{\Gamma^+}_{\phi,\psi}$.

Proof. Fix $\Gamma \in \Omega_\chi$. Remember that Γ^+ is a maximal consistent extension of Γ in $\mathcal{L}(\chi^+)$. Enumerate all the pairs of formulas (ϕ_i, ψ_i) in $\mathcal{L}(\chi)$ $\{(\phi_1, \psi_1), ..., (\phi_k, \psi_k)\}$ such that $\alpha^{\Gamma^+}_{\phi_i,\psi_i} < \beta^{\Gamma^+}_{\phi_i,\psi_i}$. Let

$$\Theta_0 = \Gamma^+,$$

and for each $n \geq 0$, let r_{n+1} be a rational number such that $r_{n+1} < \beta^{\Gamma^+}_{\phi_{n+1},\psi_{n+1}}$ and $\Theta_n \cup \{M_{r_{n+1}}(\phi_{n+1}||\psi_{n+1})\}$ is consistent, then define

$$\Theta_{n+1} = \Theta_n \cup \{M_{r_{n+1}}(\phi_{n+1}||\psi_{n+1})\}.$$

The existence of r_{n+1} with the required properties is guaranteed by Lemma 3.26. Fix Θ^∞ to be an extension of Θ_k in the language $\mathcal{L}$.

Now we prove that $\alpha^{\Theta^\infty}_{\phi_i,\psi_i} = \beta^{\Theta^\infty}_{\phi_i,\psi_i}$. If $\Diamond\psi_i \notin \Gamma$, then by Lemma 3.14 $\alpha^{\Gamma^+}_{\phi_i,\psi_i} = 1$ and $\beta^{\Gamma^+}_{\phi_i,\psi_i} = 0$, contradicting the hypothesis. Hence, $\Diamond\psi_i \in \Gamma \subset \Theta^\infty$ and by Lemma 3.17, $\alpha^{\Theta^\infty}_{\phi_i,\psi_i} = \beta^{\Theta^\infty}_{\phi_i,\psi_i}$.

To finish the proof note that for any $i > 1$ it holds that $\beta^{\Theta^\infty}_{\phi_i,\psi_i} \leq r_i$, hence

$$\beta^{\Theta^\infty}_{\phi_i,\psi_i} < \beta^{\Gamma^+}_{\phi_i,\psi_i}.$$

□

Zhou does not state Lemma 3.26 explicitly. Rather he proves the fact that Lemma 3.26 with $\Phi = \varnothing$ holds, which we call a weaker version of 3.26 for reference. Then in his paper Zhou (2009), the proof of Lemma 3.27 consists in building a series of intermediate languages for each ϕ_i such that $\alpha^{\Gamma^+}_{\phi_i} < \beta^{\Gamma^+}_{\phi_i}$. For ϕ_1 and Γ^+ the weaker version of Lemma 3.26 guarantees the existence of a r_1 such that $\Gamma^+ \cup \{M_{r_1}\phi_1\}$. Remember that $\mathcal{L}(\chi) = \mathcal{L}(P_\chi, q_\chi, d_\chi)$ and define $\mathcal{L}^1 = \mathcal{L}(P_\chi, q_1, d_\chi)$ where q_1 is the least common multiple of q_χ and the denominator of r_1. Let Γ^1 be a maximal consistent extension of Γ^+ in $\mathcal{L}^1$.

Repeat the argument for ϕ_2, i.e., if $(\beta^{\Gamma^1}_{\phi_2} < \beta^{\Gamma^+}_{\phi_2})$ or $(\beta^{\Gamma^1}_{\phi_2} = \beta^{\Gamma^+}_{\phi_2}$ and $\alpha^{\Gamma^1}_{\phi_2} = \beta^{\Gamma^1}_{\phi_2})$, then $\mathcal{L}^2 = \mathcal{L}^1$; if $\beta^{\Gamma^1}_{\phi_2} = \beta^{\Gamma^+}_{\phi_2}$ and $\alpha^{\Gamma^1}_{\phi_2} < \beta^{\Gamma^1}_{\phi_2}$, then $\mathcal{L}^2 = \mathcal{L}(P_\chi, q_2, d_\chi)$, where q_2 is the least common multiple of q_1 and the denominator of r_2 (with r_2 given by Γ^1 and the weaker version of Lemma 3.26).

Clearly the main difference between the two proofs, the one given by Lemmas 3.26 and 3.27 in this thesis and the one given by Zhou (2009) sketched above, is the fact that Lemma 3.26 is a stronger version of the idea stated by Zhou.

Now let us go back to the proof of completeness.

Enumerate all maximal consistent sets in Ω_χ, $\Gamma_1, \Gamma_2, ..., \Gamma_n$. From the above lemma it follows that for each Γ_i there is a Θ^∞_i satisfying property (F). Redefine T_χ as follows $T_\chi(\Gamma_i, [\phi]_\chi, [\psi]_\chi) = \alpha^{\Theta^\infty_i}_{\phi,\psi}$.

Define the canonical model

$$\mathcal{M}_\chi = \langle \Omega_\chi, \Sigma_\chi, T_\chi, v_\chi \rangle,$$

where $v_\chi(p) = [p]_\chi$.

Lemma 3.28 (Truth Lemma)**.** Let $\phi \in \mathcal{L}(\chi)$, then

$$\mathcal{M}_\chi, \Gamma_i \vDash \phi \text{ iff } \phi \in \Gamma_i.$$

Proof. The proof is by the complexity of ϕ, we prove the cases $\phi = L_r(\psi_1||\psi_2)$ and $\phi = \Box\psi$.

(Case $\phi = L_r(\psi_1||\psi_2)$)

Assume $\mathcal{M}_\chi, \Gamma_i \vDash L_r(\psi_1||\psi_2)$, i.e., $T_\chi(\Gamma_i, [\psi_1]_\chi, [\psi_2]_\chi) \geq r$.

If $\Diamond\chi \in \Gamma_i$ we have two cases to consider:

(Case $\alpha^{\Gamma^+_i}_{\psi_1,\psi_2} = \beta^{\Gamma^+_i}_{\psi_1,\psi_2}$) In this case $T_\chi(\Gamma_i, [\psi_1]_\chi, [\psi_2]_\chi) = \alpha^{\Gamma^+_i}_{\psi_1,\psi_2}$. Therefore $r \leq \alpha^{\Gamma^+_i}_{\psi_1,\psi_2}$ and clearly $L_r(\psi_1||\psi_2) \in \Gamma_i$.

(Case $\alpha^{\Gamma^+_i}_{\psi_1,\psi_2} < \beta^{\Gamma^+_i}_{\psi_1,\psi_2}$) In this case $T_\chi(\Gamma_i, [\psi_1]_\chi, [\psi_2]_\chi) \geq r$. Then $r \leq \alpha^{\Theta^\infty_i}_{\psi_1,\psi_2} < \beta^{\Gamma^+_i}_{\psi_1,\psi_2} = \alpha^{\Gamma^+_i}_{\psi_1,\psi_2} + \frac{1}{q_\chi}$. The last equality holds because of Lemma 3.14, but since $L_r(\psi_1||\psi_2) \in \mathcal{L}(\chi)$ we have $r \leq \alpha^{\Gamma^+_i}_{\psi_1,\psi_2}$. Hence, $L_r(\psi_1||\psi_2) \in \Gamma_i$.

If $\Diamond\chi \notin \Gamma_i$, then by Lemma 3.14, $L_1(\psi_1||\psi_2) \in \Gamma_i$, and by Theorem 3.8 it holds that $L_r(\psi_1||\psi_2) \in \Gamma_i$.

For the other direction assume $L_r(\psi_1||\psi_2) \in \Gamma_i$. Then $r \leq \alpha^{\Gamma_i^+}_{\psi_1,\psi_2} \leq \alpha^{\Theta_i^\infty}_{\psi_1,\psi_2}$. So $\mathcal{M}_\chi, \Gamma_i \models L_r(\psi_1||\psi_2)$.

(Case $\phi = \Box\psi$) Suppose $\mathcal{M}_\chi, \Gamma \vDash \Box\psi$.

Fact: If $\Box\psi \notin \Gamma$ then $U(\Gamma) \cup \{\neg\psi\}$ is consistent.

(Proof of the fact) If $U(\Gamma) \cup \{\neg\psi\}$ is inconsistent, then $\vdash \bigwedge U(\Gamma) \to \psi$, and by ($N_\Box$) and ($K_\Box$) we have that $\vdash \Box \bigwedge U(\Gamma) \to \Box\psi$. By Lemma 3.20 we also have that $\vdash \bigwedge U(\Gamma) \to \Box \bigwedge U(\Gamma)$. With the two implications $\vdash \bigwedge U(\Gamma) \to \Box\psi$ holds. Since $\Box\psi \in \mathcal{L}(\chi)$, it holds that $\Box\psi \in \Gamma$. This contradicts the hypothesis. Therefore $U(\Gamma) \cup \{\neg\psi\}$ is consistent.

Suppose $\Box\psi \notin \Gamma$. By the fact above we know that $U(\Gamma) \cup \{\neg\psi\}$ is consistent, so there is a maximal consistent set of formulas Γ' such that $U(\Gamma) \cup \{\neg\psi\} \subseteq \Gamma'$. We know that Γ' is in Ω_χ because $U(\Gamma) \subset \Gamma'$.

In this way, Γ' is such that $\Gamma \sim \Gamma'$ and $\psi \notin \Gamma'$. By the induction hypothesis $\mathcal{M}_\chi, \Gamma' \not\vDash \psi$. Then $\mathcal{M}_\chi, \Gamma \not\vDash \Box\psi$, which contradicts our assumption. Therefore $\Box\psi \in \Gamma$.

Now for the other direction assume $\Box\psi \in \Gamma$. To prove $\mathcal{M}_\chi, \Gamma \models \Box\psi$, take any $\Gamma' \in \Omega_\chi$. It suffices to show $\mathcal{M}_\chi, \Gamma' \models \psi$. Since $\Box\psi \in U(\Gamma)$ and $U(\Gamma) = U(\Gamma')$, we have $\Box\psi \in U(\Gamma') \subseteq \Gamma'$. Since $\vdash \Box\psi \to \psi$, it follows that $\psi \in \Gamma'$. By the induction hypothesis, $\mathcal{M}_\chi, \Gamma' \models \psi$. □

Theorem 3.29 (Completeness)**.** For any formula χ of $\mathcal{L}$, if χ is consistent, then there is a model $\mathcal{M} = \langle \Omega, \Sigma, T, v \rangle$, such that $\mathcal{M}, w \vDash \chi$ for some $w \in \Omega$.

An immediate consequence of the proof of this theorem is the *finite model property.* Which we state as follows.

Corollary 3.30 (Finite Model Property)**.** A formula ϕ of $\mathcal{L}$ is valid in all conditional probability models if and only if it is valid in all finite conditional probability models.

The last corollary does not imply decidability[1] because the canonical model is not finitely constructed (the values of T_χ may be irrational).

Finally, L_{cp} is not compact as shown in the next Lemma, which is the conditional probabilistic version of Theorm 3.20 in Zhou (2009).

[1]A logic is decidable if there is an algorithm that given any formula of the language decides whether it is a theorem.

Lemma 3.31 (Non-compactness)**.** L_{cp} is not compact. That is to say, there is some set Λ of formulas in $\mathcal{L}$ which is finitely satisfiable, but is not satisfiable.

Proof. Consider the set $\Lambda_\infty \doteq \{\neg L_{\frac{1}{2}} p\} \cup \{L_{\frac{1}{2}-\frac{1}{2^{n+2}}} p : n \in \mathbb{N}\}$. It is easy to see that Λ_∞ is not satisfiable. Now we show that it is finitely satisfiable. Without loss of generality, we consider the finite subset $\Lambda_N = \{\neg L_{\frac{1}{2}}\} \cup \{L_{\frac{1}{2}-\frac{1}{2^{n+2}}} p : n \leq N\}$ for some given natural number N. We define a model $\mathcal{M} = \langle \Omega, \Sigma, T, v\rangle$ as follows:

$\Omega = \{w_1, w_2\}$;

$\Sigma = \mathcal{P}(\Omega)$;

T is any Conditional Markov Kernel satisfying the condition:

$$T(w_1, \{w_1\}, \Omega) = \frac{1}{2} - \frac{1}{2^{N+3}};$$

$v(p) = \{w_1\}$.

It is easy to check that $\mathcal{M}, w_1 \vDash \Lambda_N$.

□

3.4 Completeness for Multi Agents

The proof of completeness in the multi-agent case has no added complications. To build the canonical model one has to consider a conditional Markov kernel T_a for each agent a and add the necessary steps to ensure the truth lemma for each formula of the form $L_r^a(\phi||\psi)$. In this section we will spell out some details for the multi-agent case.

Let A be a finite set of agents, let $P = \{p, q, ...\}$ be the set of propositional variables and consider the language obtained from the following:

$$\phi \doteq \top|p|\neg\phi|\phi \wedge \phi|\Box\phi|L_r^a(\phi||\phi), \tag{3.2}$$

where $p \in P, a \in A$ and $r \in [0,1] \cap \mathbb{Q}$.

Read $L_r^a p$ as 'agent a believes the probability of p is at least r.' Define the following abbreviation:

$$L_{1-r}^a(\neg\phi||\psi) \doteq M_r^a(\phi||\psi).$$

A *multi-agent conditional probability model* $\mathcal{M} = \langle \Omega, \Sigma, (T_a)_{a\in A}, v\rangle$ is such that (Ω, Σ) is a measurable space; $T_a : \Omega \times \Sigma \times \Sigma \to [0,1]$ is a conditional Markov kernel for each $a \in A$; v is a valuation, a function from the set of propositional letters to Σ.

The set of axioms has the axioms $(A1)$ to $(A5), (C1), (C2)$ and $(C_{\rightarrow})$ and the rule $(ARCH)$ for each agent $a \in A$.

For completeness define the *depth* of a formula as previously, i.e, if ϕ is a non-probabilistic formula, then $dp(\phi)$ is like before, while the depth of a probabilistic formula is given by $dp(L_r^a(\psi||\chi)) = 1 + \max\{dp(\psi), dp(\chi)\}$.

With this definition of depth fix a consistent formula χ and define the local language $\mathcal{L}(\chi) = \mathcal{L}(P_\chi, d_\chi, q_\chi)$.

Let Γ_χ be a maximal consistent set of formulas of $\mathcal{L}(\chi)$ that contains χ. Define $\Omega_\chi = \{\Delta \in \Omega : \Delta \sim \Gamma_\chi\}$, where $\sim$ is defined as before. Let the σ-algebra be $\Sigma_\chi = \{[\phi]_\chi : \phi \in \mathcal{L}(\chi)\}$.

To construct the conditional Markov kernel T_a we define two functions α and β. For each $\Gamma \in \Omega_\chi$, fix an extension $\Gamma^+ \in \mathcal{L}(\chi^+)$. Define

$$\alpha(a, \Gamma, \phi, \psi) = \max\{r : L_r^a(\phi||\psi) \in \Gamma^+\} \text{ and } \beta(a, \Gamma, \phi, \psi) = \min\{r : M_r^a(\phi||\psi) \in \Gamma^+\}.$$

For each extension Γ^+ (of Γ) fix a maximal consistent extension Γ^∞ in $\mathcal{L}$ and define:

$$\alpha(a, \Gamma^\infty, \phi, \psi) = sup\{r : L_r^a(\phi||\psi) \in \Gamma^\infty\} \text{ and } \beta(a, \Gamma^\infty, \phi, \psi) = inf\{r : M_r^a(\phi||\psi) \in \Gamma^\infty\}.$$

The key lemma here is Lemma 3.27, of which we state a modified version below.

Lemma 3.32. For each Γ in Ω_χ there is a maximal consistent extension Θ^∞ of Γ^+ in $\mathcal{L}$ such that the following property is satisfied:

(F_m): for each agent a for any $\phi, \psi \in \mathcal{L}(\chi)$, if $\alpha(a, \Gamma, \phi, \psi) < \beta(a, \Gamma, \phi, \psi)$, then

$$\alpha(a, \Gamma, \phi, \psi) \leq \alpha(a, \Theta^\infty, \phi, \psi) = \beta(a, \Theta^\infty, \phi, \psi) < \beta(a, \Gamma, \phi, \psi).$$

Proof. Fix $\Gamma \in \Omega_\chi$. Remember that Γ^+ is a maximal consistent extension of Γ in $\mathcal{L}(\chi^+)$. Enumerate all the triples (a_i, ϕ_i, ψ_i) with $a_i \in A$ and $\phi_i, \psi_i \in \mathcal{L}(\chi)$:
$\{(a_i, \phi_1, \psi_1), ..., (a_k, \phi_k, \psi_k)\}$ such that $\alpha(a_i, \Gamma^+, \phi_i, \psi_i) < \beta(a_i, \Gamma^+, \phi_i, \psi_i)$. Note that the agent a_i does not correspond to the i-th agent in the set A. Let

$$\Theta_0 = \Gamma^+,$$

and for each $n \geq 0$, let r_{n+1} be a rational number such that $r_{n+1} < \beta(a_{n+1}, \Gamma^+, \phi_{n+1}, \psi_{n+1})$ and $\Theta_n \cup \{M^a_{r_{n+1}}(\phi_{n+1}||\psi_{n+1})\}$ is consistent, then define

$$\Theta_{n+1} = \Theta_n \cup \{M^a_{r_{n+1}}(\phi_{n+1}||\psi_{n+1})\}.$$

The existence of r_{n+1} with the required properties is guaranteed by Lemma 3.26. Fix Θ^∞ to be an extension of Θ_k in the language $\mathcal{L}$.

Now we prove that for each $a \in A$, $\alpha(a, \Theta^\infty, \phi, \psi) = \beta(a, \Theta^\infty, \phi, \psi)$. If $\Diamond\psi \notin \Gamma$, then by Lemma 3.14, $\alpha(a, \Theta^\infty, \phi, \psi) = 1$ and $\beta(a, \Theta^\infty, \phi, \psi) = 0$, contradicting the hypothesis. Hence, $\Diamond\psi \in \Gamma \subset \Theta^\infty$ and by Lemma 3.17, $\alpha(a, \Theta^\infty, \phi, \psi) = \beta(a, \Theta^\infty, \phi, \psi)$.

To finish the proof note that for any $i > 1$ it holds that $\beta(a_i, \Theta^\infty, \phi_i, \psi_i) \leq r_i$, hence

$$\beta(a_i, \Theta^\infty, \phi_i, \psi_i) < \beta(a_i, \Gamma^+, \phi_i, \psi_i).$$

□

Enumerate all maximal consistent sets in Ω_χ, $\Gamma_1, \Gamma_2, ..., \Gamma_n$. From the above lemma it follows that for each Γ_i there is a Θ_i^∞ satisfying property (F_m). Define T_a as follows $T_a(\Gamma_i, [\phi]_\chi, [\psi]_\chi) = \alpha(a, \Theta_i^\infty, \phi, \psi)$.

Chapter 4

Dynamics

> Ela acreditava em anjo e, porque acreditava, eles existiam.
>
> — Clarice Lispector

> She did believe in angel and, because she believed, they existed.
>
> — Clarice Lispector

The main point of this chapter is to extend conditional probability logic with dynamic operators. With that we express the flow of information relevant for, among other things, card games. The change of information is caused by a variety of ways, some of the ones that we describe in the following sections are announcements, turning cards face up, drawing cards, etc. This flow and change of information is traditional known as dynamic epistemic logic.

Dynamic epistemic logic is well studied in the book by van Ditmarsch et al. (2007), but it did not treat probabilistic cases of dynamics. The papers by Sack (2009) and then by Kooi (2003) were the first to treat the probability case. However, they were using as model the traditional probability spaces, hence formulas that were believed to be false could not be announced and then begin to be believed.

In this chapter we refer to event models for probabilistic models that were well studied in van Benthem (2003) and improved in van Benthem et al. (2009). Our approach in this chapter is different from the ones proposed in those papers in two points: 1. We make use of conditional probability models; and 2. our language is inspired by Aumann's idea, as explained in the introduction, while van Benthem at al. are inspired by Halpern's ideas. Notice that the language we propose for conditional probability models is not a sublanguage of Halpern's language. The advantage of allowing formulas to express inequalities and sum of probabilities is clear with the proof of completeness through a set of reduction axioms for update models. With our language we could mimic only part of this result.

Also in this chapter we refer to Dynamic Epistemic Logic with assignment, which is studied in Benthem et al. (2006), Kooi (2007) and van Ditmarsch et al. (2005) without considering probabilistic beliefs. In this chapter we extend those ideas to accommodate conditional probability epistemic logics.

In this chapter one of the reasons for introducing conditional probabilistic language in Chapter 3 becomes clear. We start with public announcement logic as a case study and as a natural motivation for the introduction of conditional probabilistic logic.

4.1 Why Conditional Probability Logic, via PAL

One strength of Renyi's notion of conditional probability is the meaningfulness of conditionalizing over events of probability zero. An application of that for logic is the possibility of expressing the following idea: someone believes in something to be false when it is actually true, and when faced with the facts his/her belief in that something is greater than zero. We spell out this notion formally with the help of public announcement logic in the next example.

Example 4.1. Suppose that today You and I are in a room without windows and that it is raining outside but I do not believe it. In fact, I believe that the chance that it is not raining today is 100%. I believe that when it is raining You have an umbrella 70% of the time. And I believe that when it is not raining you carry the umbrella 40% of the time. It is very clear that without anything else I believe that the chance that You have an umbrella today is 40% (but in fact the probability is 70% as it is raining but I do not believe it).

It should also be clear that after you telling me that it is raining I should believe that the chance that You have an umbrella today is 70%. But how to model that?

Let us restate the problem with a static conditional probabilistic model.

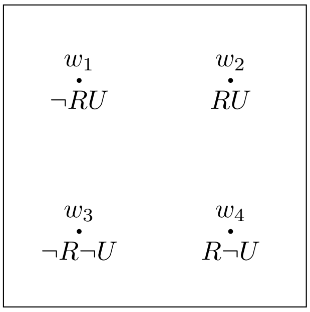

Figure 4.1: Example 4.1. R stands for 'it is raining' and U for 'You have an umbrella.'

Let $\mathcal{M} = \langle \Omega, \Sigma, T, v \rangle$ be the following model with any function T satisfying the equalities below:

$\Omega = \{w_1, w_2, w_3, w_4\}$

$\Sigma = \mathcal{P}(\Omega)$

$T(x, [\![U]\!], [\![R]\!]) = 0.7$, $T(x, [\![U]\!], [\![\neg R]\!]) = 0.4$ and $T(x, [\![\neg R]\!], \Omega) = 1$, for all $x \in \Omega$.

$v(R) = \{w_2, w_4\}$, $v(U) = \{w_1, w_2\}$.

Where R reads as 'it is raining' and U as 'You have an umbrella.' See figure 4.1 for a picture of the model.

For the information change we use the notion of announcements. The announcement of ϕ is essentially a set of states (the states where the announcement is true). If ϕ is a formula, read $[!\phi]\psi$ as 'after the announcement of ϕ it is the case that ψ'. The language of *probabilistic* PAL is given as:

$$\phi = p|\neg\phi|\phi \wedge \phi|L_r(\phi||\phi)|\Box\phi|[!\phi]\phi$$

Given a conditional probability model $\mathcal{M} = \langle \Omega, \Sigma, T, v \rangle$ the definition of truth of formulas in probabilistic PAL is like before, with the added clause:

$$\mathcal{M}, w \vDash [!\phi]\psi \text{ iff } \mathcal{M}, w \vDash \phi \text{ implies } \mathcal{M}_\phi, w \vDash \psi;$$

where $\mathcal{M}_\phi = \langle \Omega_\phi, \Sigma_\phi, T_\phi, v_\phi \rangle$ is defined as follows:

$\Omega_\phi = [\![\phi]\!]$;

$\Sigma_\phi = \{B \cap \Omega_\phi : B \in \Sigma\}$;

$T_\phi(w, A, B) = T(w, A, B)$;

$v_\phi(\cdot) = v(\cdot) \cap \Omega_\phi$

Define $[\![\psi]\!]_\phi \doteq \{w \in \Omega_\phi : \mathcal{M}_\phi, w \vDash \psi\}$.

Let us go back to Example 4.1 and see the definition of public announcement in action.

Example 4.2 (Continuation of Example 4.1)**.** Given the model $\mathcal{M}$ in Example 4.1 the following facts are true:

1. $\mathcal{M} \vDash L_1 \neg R$
2. $\mathcal{M} \vDash L_{0.4} U \wedge M_{0.4} U$
3. $\mathcal{M} \vDash [!R] L_{0.7} U$
4. $\mathcal{M} \vDash [!R] L_1 R$

Proof. We prove only 2 and 3.

2. We have to prove that $T(w, [\![U]\!], \Omega) = 0.4$, for any $w \in \Omega$. First fix $w \in \Omega$ and remember that, since $\Omega \neq \varnothing$, then $T(x, [\![U]\!], \Omega) = T(w, [\![U \wedge R]\!], \Omega) + T(w, [\![U \wedge \neg R]\!], \Omega)$. since $T(w, \cdot, \Omega)$ is a conditional probability function we have the equalities:

(a) $T(w, [\![U \wedge \neg R]\!], \Omega) = T(w, [\![U]\!], [\![\neg R]\!] \cap \Omega) \cdot T(w, [\![\neg R]\!], \Omega) = 0.4 \cdot 1 = 0.4$.

(b) $T(w, [\![U \wedge R]\!], \Omega) = T(w, [\![U]\!], [\![R]\!] \cap \Omega) \cdot T(w, [\![R]\!], \Omega) = 0.7 \cdot 0 = 0$

Therefore $T(w, [\![U]\!], \Omega) = 0.4$ for any $w \in \Omega$, which implies $\mathcal{M} \vDash L_{0.4} U \wedge M_{0.4} U$.

3. Denote by Ω_R the support of $\mathcal{M}_R$, the model resulted from the announcement of R. Note that $\Omega_R = \Omega \cap [\![R]\!]$. And similarly $[\![U]\!]_R = [\![U \wedge R]\!]$. Finally remember $T(w, [\![U \wedge R]\!], [\![R]\!]) = T(w, [\![U]\!], [\![R]\!])$ for any U and R.

With this we have:

$\mathcal{M} \vDash [!R] L_{0.7} U$ if only if $\mathcal{M} \vDash R$ implies $\mathcal{M}_R \vDash L_{0.7} U$, which holds if and only if $\mathcal{M} \vDash R$ implies $T_R(w, [\![U]\!]_R, \Omega_R) \geq 0.7$ for all w, which holds. □

Examples 4.1 and 4.2 illustrate the strength of conditional probability models. While it is intuitive that agents should review their beliefs when faced with information contradicting previous ones, it is not clear how this update should be expressed without conditional probability spaces.

The reason for introducing a complete set of axioms for conditional probability logic is that there is no reduction axioms for PAL in probability logic based in Aumann's language. If p and q are propositional variables we can easily see that the following is a valid formula:

$$[!p] L_r q \leftrightarrow (p \rightarrow L_r(q || p)),$$

i.e., from a formula without conditional probability we end up with a formula with conditional probability.

The power of conditional probability logic goes beyond PAL, but as an introduction to the topic we prove the completeness of PAL for conditional probability logic through reduction axioms.

Lemma 4.3. Let ϕ and ψ be formulas and $\mathcal{M}$ a model for probabilistic PAL, then:

$$[\![\psi]\!]_\phi = [\![\phi \wedge [!\phi]\psi]\!]$$

Proof. Suppose $w \in [\![\phi \wedge [!\phi]\psi]\!]$. Then $\mathcal{M}, w \vDash \phi$ and ($\mathcal{M}, w \vDash \phi$ implies $\mathcal{M}_\phi, w \vDash \psi$). Therefore $\mathcal{M}_\phi, w \vDash \psi$, which is the same as $w \in [\![\psi]\!]_\phi$.

Suppose $w \in [\![\psi]\!]_\phi$, i.e., $w \in [\![\phi]\!]$ and $\mathcal{M}_\phi, w \vDash \psi$. This is the same as to say that $\mathcal{M}, w \vDash \phi$ and ($\mathcal{M}, w \vDash \phi$ implies $\mathcal{M}_\phi, w \vDash \psi$). Which is exactly $w \in [\![\phi \wedge [!\phi]\psi]\!]$. □

We prove the completeness of probabilistic PAL by giving a set of reduction axioms. Reduction axioms are schemata of formulas which together describe a procedure to replace a formula with the connective $[!\phi]$ by an equivalent formula without the connective.

Theorem 4.4. The reduction axioms for probabilistic PAL, given by the schemata of formulas below, are valid.

$\cdot [!\phi]p \leftrightarrow (\phi \rightarrow p)$
$\cdot [!\phi]\neg\psi \leftrightarrow (\phi \rightarrow \neg[!\phi]\psi)$
$\cdot [!\phi]\Box\psi \leftrightarrow (\phi \rightarrow \Box[!\phi]\psi)$
$\cdot [!\phi](\psi \wedge \chi) \leftrightarrow ([!\phi]\psi \wedge [!\phi]\chi)$
$\cdot [!\phi]L_r(\psi||\chi) \leftrightarrow (\phi \rightarrow L_r([!\phi]\psi||\phi \wedge [!\phi]\chi))$

Proof. We prove only the probabilistic formula. The others are routine.

Let $\mathcal{M}$ be a conditional probability model and let w be a state in the model.

$\mathcal{M}, w \vDash [!\phi]L_r(\psi||\chi)$ iff $\mathcal{M}, w \vDash \phi$ implies $T_\phi(w, [\![\psi]\!]_\phi, [\![\chi]\!]_\phi) \geq r$, which is equivalent, by the previous lemma, to:

$$\mathcal{M}, w \vDash \phi \text{ implies } T(w, [\![\phi \wedge [!\phi]\psi]\!], [\![\phi \wedge [!\phi]\chi]\!]) \geq r.$$

Remember that $T(w, A \cap B, A \cap C) = T(w, B, A \cap C)$ always hold (see Theorem 2.2), then

$$T(w, [\![[!\phi]\psi]\!], [\![\phi \wedge [!\phi]\chi]\!]) = T(w, [\![\phi \wedge [!\phi]\psi]\!], [\![\phi \wedge [!\phi]\chi]\!]).$$

So the above condition is equivalent to $\mathcal{M}, w \models \phi \rightarrow L_r([!\phi]\psi||\phi \wedge [!\phi]\chi)$. □

The last reduction axiom cannot be expressed in probabilistic logic. We can see that the conditional probability operator appears in the reduction schema for the non-conditional probability operator if we take $\chi = \top$ in the last reduction axiom:

$$[!\phi]L_r(\psi) \leftrightarrow \phi \to L_r([!\phi]\psi||\phi).$$

The ability to express this equivalence is one of the advantages of conditional probability logic over the standard probability logic of Zhou (2009).

Theorem 4.5. The system L_{cp} together with the reduction axioms in Theorem 4.4 and the rule

$$\vdash \phi \leftrightarrow \phi' \text{ imples } \vdash [!\psi]\phi \leftrightarrow [!\psi]\phi'$$

are a complete system for probabilistic PAL.

Proof. Consider the following translation:

$$\begin{aligned}
\overline{p} &= p \\
\overline{\neg\phi} &= \neg\overline{\phi} \\
\overline{\Box\phi} &= \Box\overline{\phi} \\
\overline{\phi \wedge \psi} &= \overline{\phi} \wedge \overline{\psi} \\
\overline{L_r(\phi||\psi)} &= L_r(\overline{\phi}||\overline{\psi}) \\
\overline{[!\phi]p} &= \overline{\phi} \to p \\
\overline{[!\phi]\neg\psi} &= \overline{\phi} \to \neg\overline{[!\phi]\psi} \\
\overline{[!\phi]\Box\psi} &= \overline{\phi} \to \Box\overline{[!\phi]\psi} \\
\overline{[!\phi](\psi_1 \wedge \psi_2)} &= \overline{[!\phi]\psi_1} \wedge \overline{[!\phi]\psi_2} \\
\overline{[!\phi]L_r(\psi_1||\psi_2)} &= \overline{\phi} \to L_r(\overline{[!\phi]\psi_1}||\overline{\phi \wedge [!\phi]\psi_2}) \\
\overline{[!\phi_1][!\phi_2]\psi} &= \overline{[!\phi_1](\overline{[!\phi_2]\psi})}
\end{aligned}$$

The translation $\overline{\phi}$ is defined on all formulas of PAL and the formula $\overline{\phi}$ does not contain any occurrence of the operator $[!\cdot]$. Moreover, one can prove by induction on the complexity of ϕ that

$$\vdash \phi \leftrightarrow \overline{\phi}.$$

Suppose ϕ is a formula in the probabilistic PAL such that it is valid in all models in the class of conditional probability models. Hence, $\overline{\phi}$ is also valid and by the completeness of conditional probability logic (Theorem 3.29) there is a proof of $\overline{\phi}$. Therefore, since $\vdash \phi \leftrightarrow \overline{\phi}$ is provable, there is a proof of ϕ. ∎

To sum up the discussion in this section we state the following theorem:

Theorem 4.6. The formula $L_1\neg p \wedge p \wedge [!p]L_r p$ for $r > 0$ is satisfiable.

Proof. The model that satisfies the formula is given in Example 4.1 and the proof of satisfiability is given by Example 4.2. □

4.2 Events

In this section we introduce a broader class of mechanisms to update a model. We call a given change in the world an *event* and present as examples announcements, the turning of a lamp on or off, the turning of a card face up, the drawing of a card, etc.

The word 'event' from event models should not be confused with events of a probability space. The first refers to a change in the world (either epistemic or factual); the second is a set of possible outcomes in a random process.

A worthwhile remark about this issue is made in a footnote in van Benthem et al. (2009). We reproduce the footnote below:

> It is rather unfortunate that the term *event* is widely used in both probability theory and dynamic-epistemic logic, but with slightly different interpretations. In probability theory an event is what one would call a proposition in logic. While an event in dynamic epistemic logic also comes with a proposition, viz. its precondition, events in an event model really transform a given probabilistic model, and are not part of that model itself. To make matters worse, sometimes a whole event model is referred to as an event in dynamic-epistemic logic. We can only warn the reader to suspend any easy identifications across fields here.

Next first we introduce assignments, events that change the state of the world (factual changes), then we introduce assignments with preconditions. Finally, we introduce event assignment models and product update for sets of events which the agent is unsure about which one took place.

4.2.1 Assignments

Another way to modify a given scenario is by performing factual changes in the world. In this section we will introduce the notion of assignments without preconditions, events that simply change the state of the world.

Example 4.7. Consider a lamp that turns on or off though a sensor as follows: if the lamp is on and the sensor is activated the lamp goes off, and if it is off and the sensor is activated the lamp turns on.

States in which the lamp is on are elements of $[\![ON]\!]$ and we call them ON states. Similarly, states in which the lamp is off are elements of $[\![OFF]\!]$ and we call them OFF states. Every time the lamp's sensor is activated all the ON states becomes OFF states and vice-versa. Note that the action of activating the sensor has no precondition and can always be performed.

We denote this change by $[ON := OFF, OFF := ON]$ and call an assignment. The general case is given in the next definition.

Definition 4.8. An *assignment* τ is a finite sequence of the form $([p_1 := \phi_1], ..., [p_n := \phi_n])$ where each propositional variable p appears at most once on the left side. We sometimes denote ϕ_i by ϕ_{p_i}. An assignment τ can be seen as a finite partial function from the set of propositional variables to the set of formulas:

$$\tau(p) = \begin{cases} \phi_p & \text{if } p \text{ is in the domain of } \tau \\ p & \text{otherwise.} \end{cases}$$

We extend the assignment function to the whole language distributing it through the connectives, e.g., $\tau(\neg\phi) = \neg\tau(\phi), \tau(L_r(\phi||\psi)) = L_r(\tau(\phi)||\tau(\psi))$, etc.

Probabilistic Assignment Logic is given by

$$\phi \doteq\ p \mid \neg\phi \mid \phi \wedge \phi \mid L_r(\phi||\psi) \mid \Box\phi \mid [\tau]\phi$$

where p is a propositional variable and τ is an assignment.

Given a conditional probability model $\mathcal{M} = \langle \Omega, \Sigma, T, v\rangle$ we define $\mathcal{M}_\tau = \langle \Omega, \Sigma, T, v'\rangle$ where $v'(p) = [\![\tau(p)]\!]_{\mathcal{M}}$. The definition of truth is given as before with the added clause:

$$\mathcal{M}, w \vDash [\tau]\phi \text{ iff } \mathcal{M}_\tau, w \vDash \phi$$

.

Example 4.9 (Revisiting Example 4.7)**.** Consider the following conditional probabilistic model for Example 4.7, $\mathcal{M} = \langle \Omega = \{w_1, w_2\}, \mathcal{P}(\Omega), T, v\rangle$ where T is given by $T(w, \{w_1\}) = \frac{2}{3}$ and $T(w, \{w_2\}) = \frac{1}{3}$ for any $w \in \Omega$ and $v(ON) = \{w_1\}$ and $v(OFF) = \{w_2\}$, i.e., w_1 is an ON state and w_2 is an OFF state.

In this situation the agent believes it is twice more likely that the lamp is on than it is off, i.e., it holds that $\mathcal{M} \vDash L_{\frac{1}{3}} OFF \wedge L_{\frac{2}{3}} ON$.

Let $\tau_{sen} = [ON := OFF, OFF := ON]$ be the assignment related to activating the sensor of the lamp. We can see that after the lamp is activated the agent's beliefs change, i.e., $\mathcal{M} \vDash [\tau_{sen}] L_{\frac{2}{3}} OFF \wedge L_{\frac{1}{3}} ON$. The proof goes as follows:

$T(w, [\![ON]\!]) \geq \frac{2}{3}$ for all $w \in \Omega$ implies that $T(w, [\![\tau_{sen}(OFF)]\!]) \geq \frac{2}{3}$ for all $w \in \Omega$. This implies that $\mathcal{M}_{\tau_{sen}} \vDash L_{\frac{2}{3}} OFF$ and hence $\mathcal{M} \vDash [\tau_{sen}] L_{\frac{2}{3}} OFF$.

For $\mathcal{M} \vDash [\tau_{sen}] L_{\frac{1}{3}} ON$ the reasoning is similar. Also note that it is valid that

$$\mathcal{M} \vDash [\tau_{sen}](\phi \wedge \psi) \leftrightarrow ([\tau_{sen}]\phi \wedge [\tau_{sen}]\psi).$$

Moreover, this equivalence is valid for any assignment in the class of the conditional probability spaces. We prove that in our next theorem.

Conditional probability logic extended with assignments is complete with respect to the class of conditional probability spaces. Furthermore, its completeness is given by a set of reduction axioms.

Lemma 4.10. The following equation holds:

$$[\![\phi]\!]_{\mathcal{M}_\tau} = [\![\tau(\phi)]\!]_{\mathcal{M}} = [\![[\tau]\phi]\!]_{\mathcal{M}}$$

Proof. The proof is by induction on the complexity of ϕ. □

Theorem 4.11. The following formulas are valid:

$\cdot [\tau]p \leftrightarrow \tau(p)$

$\cdot [\tau]\neg\psi \leftrightarrow \neg[\tau]\psi$

$\cdot [\tau]\Box\psi \leftrightarrow (\Box[\tau]\psi)$

$\cdot [\tau](\psi \wedge \chi) \leftrightarrow ([\tau]\psi \wedge [\tau]\chi)$

$\cdot [\tau]L_r(\psi || \chi) \leftrightarrow L_r([\tau]\psi || [\tau]\chi)$

We postpone the proof until the next section where we introduce assignments with preconditions.

It is worth noting that the formula $[\tau]L_r(\phi||\top) \leftrightarrow L_r([\tau]\phi||\top)$ is a valid formula. This means that we could have a set of reduction axioms for (non conditional) probability logic extended with assignments.

Assignments without preconditions like defined in this section are the simplest kind of events we consider, but because of their simplicity some events like the draw or turn of a card in Algo, for example, are not possible to model. In order to express more complex changes in the state of the world we introduce the notion of assignments with preconditions in the next section.

4.2.2 Assignments with preconditions

A precondition of an event is a formula in conditional probability logic. Given a conditional probability model, the actual world is an element of the set of states that satisfies the precondition associated to the event before the event has taken place (otherwise the event could not have taken place). Essentially, the precondition of an event represents the set of states in which the event can happen. If the event happened, we should exclude states that are inconsistent with the event taking place.

We formalize this discussion in this section.

Definition 4.12. An *assignment-precondition* event e is a pair (τ, pre), where τ is an assignment and pre is a conditional probability logic formula, the precondition of τ. We sometimes denote by τ_e and $pre(e)$ the precondition associated with the pair $e = (\tau, pre)$.

We define the language for assignment-precondition probabilistic logic by:

$$\phi \doteq \ p \mid \neg\phi \mid \phi \wedge \phi \mid L_r(\phi||\phi) \mid \Box\phi \mid [e]\phi$$

where p is a propositional variable and e is an assignment-precondition event.

The satisfaction relation is given as before with the added clause:

$$\mathcal{M}, w \vDash [e]\phi \text{ iff } \mathcal{M}, w \vDash pre(e) \text{ implies } \mathcal{M}_e, w \vDash \phi,$$

where $\mathcal{M}_e = \langle \Omega_e, \Sigma_e, T_e, v_e \rangle$ with

· $\Omega_e = [\![pre(e)]\!]_{\mathcal{M}}$

· $\Sigma_e = \{B \cap \Omega_e : B \in \Sigma\}$

· $T_e(w, A, B) = T(w, A, B)$

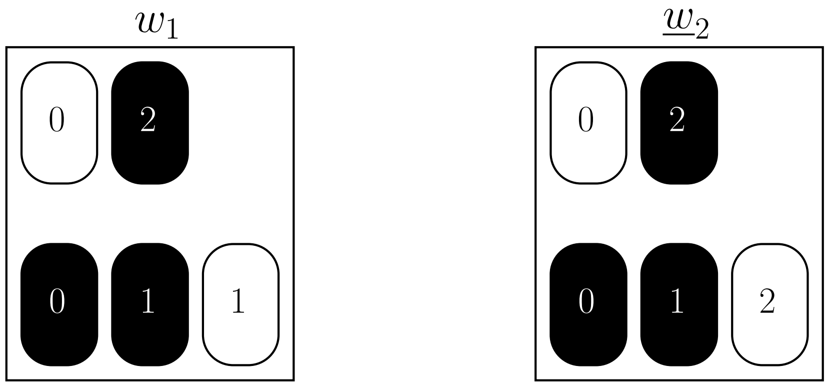

Figure 4.2: Example 4.13. Player 2 cannot distinguish between w_1 and w_2.

· $v_e(p) = [\![\tau_e(p)]\!]_{\mathcal{M}}$

Let's consider the turn of a card.

Example 4.13 (Turning a card face up)**.** Consider the following situation of Algo$_2$ as follows: player 2 has the cards $(0W, 2B)$ and both are turning face down. Player 1 has the cards $(0B, 1B, 2W)$ and all the cards are turning face down. Remember that the cards in Algo$_2$ ranges from 0 to 2.

Let w_1 be the state in which player 1 holds the cards $(0B, 1B, 1W)$ and w_2 be the state in which player 1 holds the card $(0B, 1B, 2W)$. Note that player 2 believes it is equally likely that the actual state is either w_1 or w_2 despite the fact that the actual state is w_2. See Figure 4.2.

For the given situation we have the following conditional probability model $\mathcal{M}$:

· $\Omega = \{w_1, w_2\}$

· $\Sigma = \mathcal{P}(\Omega)$

For all $x \in \Omega$:

· $T(x, \{w\}, \Omega) = \frac{1}{2}$ for all $w \in \Omega$ and $T(x, \{w\}, \{u\}) = 1$ if $w = u$; 0 otherwise

· $v(1W1_3) = \{w_1\}$, $v(2W1_3) = \{w_2\}$ and $v(3W1_3) = \varnothing$.

Let $L_r p$ stand for "player 2 believes the chance that p is true is at least r." Clearly it holds that $\mathcal{M} \vDash L_{\frac{1}{2}}(1W1_1)$.

Let e_1 be the assignment-precondition event of turning the card $1W$ in the third position in player 1's hand face up. Similarly, let e_2 be the assignment-precondition event of turning the card $2W$ in the third in player 1's hand face up. Formally we have

$e_i = (\tau, iW1_3 \wedge D_{1_i})$ for $i \leq 2$ and $\tau = \{D_{1_3} := U_{1_3}\}$. Remember that $D_{1_3}(U_{1_3})$ stands for 'the card in position 3 in the hand of player 1 is face down(up).'

Clearly, after card $2W$ is turned face up player 2 is 100% sure of player 1's hand. We can see that by $\mathcal{M}, w_2 \vDash [e_2]L_1(2W1_3)$. On the other hand, if the card turned was $1W$ player 2 would be also sure about player 1's hand configuration. We can see that by $\mathcal{M}, w_2 \vDash [e_1]L_1(1W1_3)$. This last relation is true because w_2 does not satisfy the precondition of e_1.

Finally we can also say that after the turn of the card in the third position in player 1's hand player 2 would be 100% sure about player 1's hand, i.e.,

$$\mathcal{M} \vDash [e_1]L_1(1W1_3) \wedge [e_2]L_1(2W1_3)$$

holds.

There is a set of reduction axioms for conditional probabilistic logic extended with assignment-precondition connective as we see in our next theorem.

Lemma 4.14. The following equality holds:

$$[\![\phi]\!]_{\mathcal{M}_e} = [\![pre(e) \wedge [e]\phi]\!]_{\mathcal{M}}$$

Theorem 4.15. The following formulas are valid for any assignment-precondition e:

$\cdot[e]p \leftrightarrow (pre(e) \rightarrow p)$

$\cdot[e]\neg\psi \leftrightarrow (pre(e) \rightarrow \neg[e]\psi)$

$\cdot[e]\Box\psi \leftrightarrow (pre(e) \rightarrow \Box[e]\psi)$

$\cdot[e](\psi \wedge \chi) \leftrightarrow ([e]\psi \wedge [e]\chi)$

$\cdot[e]L_r(\phi||\psi) \leftrightarrow (pre(e) \rightarrow L_r([e]\phi||pre(e) \wedge [e]\psi))$

Proof. We prove the last equivalence:

$\mathcal{M}, w \vDash [e]L_r(\phi||\psi)$ iff $\mathcal{M}, w \vDash pre(e)$ implies $\mathcal{M}_e, w \vDash L_r(\phi||\psi)$.

Note that $\mathcal{M}_e, (w, e) \vDash L_r(\phi||\psi)$ is equivalent to $T(w, [\![\phi]\!]_{\mathcal{M}_e}, [\![\psi]\!]_{\mathcal{M}_e}) \geq r$.

The following holds because of Lemma 4.14.

$$T(w, ([\![\phi]\!]_{\mathcal{M}_e}), ([\![\psi]\!]_{\mathcal{M}_e})) = T(w, [\![[e]\phi]\!], [\![pre(e) \wedge [e]\psi]\!]) \geq r,$$

which turns out to be equivalent to

$$\mathcal{M}, w \vDash L_r([e]\phi||pre(e) \wedge [e]\psi).$$

□

4.2.3 Event assignment model and product update model

Until now we considered events happening by themselves. In those cases the agent could be sure that an event was happening and he/she knew which event took place. Now we want to model situations where the agent knows that some event took place but cannot distinguish which particular one.

In this section we define the product model resulted from a conditional probability model and an event assignment model. An event assignment model is essentially a set of events (with assignments) with a probability distribution over it, representing the uncertainty of which event took place. Let's consider an example first.

Example 4.16. Consider the following state of an Algo game: Player 1's hand is $(1B, 3W{,}5W, 6B, 10W)$, the last car drawn was $1B$ and Player 2's hand is $(0W, 5B, 9B, 11W)$. Player 1 guesses $5B$, $11W$ and mistakenly guesses $8B$ in the third position in player 2's hand. Now it is the turn of player 2 and she draws the card $8B$, which is put at position 3. See Figure 4.3 for the draw action.

In this situation player 1 cannot distinguish between the events where player 2 draws the card $7B$ and the actual card $8B$. In fact, player 1 cannot distinguish between the cards $7B, 8B, 9B$ or $10B$.

In this section we will formalize the following discussion: Suppose that before the draw the formula $L^1_r(7B2_3)$ holds, i.e., player 1 believes that the chance that the third card $7B$ is in the third position in player 2's hand is at least r. Hence, after the draw the formula $M^1_0(7B2_4)$ should hold, i.e., player 1 considers impossible that the the forth card in player 2's hand is a $7B$ (assuming a normal distribution over the possible draws).

Another idea that we want to formalize is that if the formula $L^1_r(8B2_3)$ holds before the draw, then it should hold that $L^1_{r'}(8B2_4)$ for some r' after. How to calculate r' is also of interest in this section.

We start with the definition of event assignment model in a multi-agent scenario. Let A be a non-empty set of agents:

Definition 4.17. An *event assignment model* is a tuple $\mathcal{E} = \langle E, pre, (P_a)_{a\in A}, \tau\rangle$ where

- E is a finite set of events,
- $pre(e)$ is the precondition for event e; a precondition is a formula of conditional probability logic.
- $P_a(e, \cdot)$ is a probability distribution over E for each agent a in A.

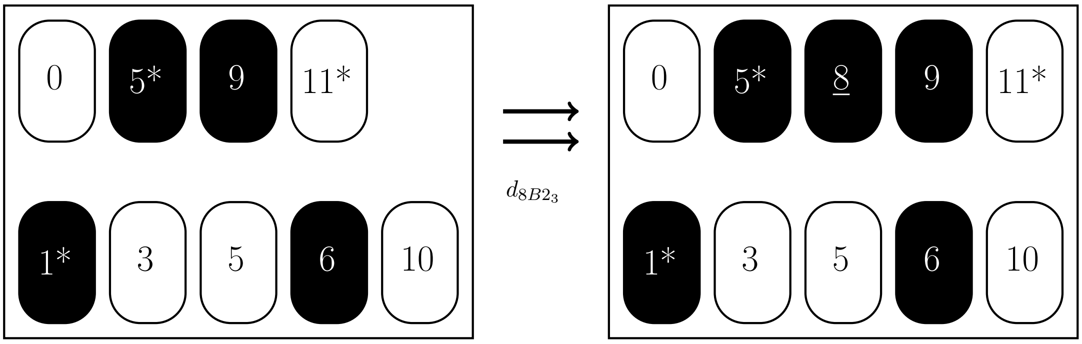

Figure 4.3: Example 4.16. The hand of Player 2 is $(0W, 5B, 9B, 11W)$. The hand of Player 1 is $(1B, 3W, 5W, 6B, 10W)$. The asterisk '$(*)$' represents the cards that are face up and the underline ('$\underline{8}$') represents the last card drawn.

· $\cdot \tau_e$ is an assignment, the change cause by event e.

The probability $P_a(e, \{e'\})$ represents the probability the agent a assigns to certain event e' happening when the event e actually happens.

It is worth mentioning that van Benthem et al. (2009) included a third "occurrence probability" which is a probability distribution on the events sharing the same precondition. We avoid that scenario here.

We define the language for *event conditional probability logic* (E-CPL henceforth) as follows:

$$\phi \doteq p \mid \neg\phi \mid \phi \wedge \phi \mid L^i_r(\phi||\psi) \mid \Box\phi \mid [\mathcal{E}, e]\phi$$

where p is a propositional variable, $i \in A$, $r \in [0,1] \cap \mathbb{Q}$ and $\mathcal{E}$ is an event assignment model.

Definition 4.18. Given a conditional probabilistic model $\mathcal{M} = \langle \Omega, \Sigma, (T_i)_{i \in A}, v \rangle$ and an event assignment model $\mathcal{E} = \langle E, pre, (P_a)_{a \in A}, \tau \rangle$ we define truth as before with the added clause:

$$\mathcal{M}, w \vDash [\mathcal{E}, e]\phi \text{ iff } \mathcal{M}, w \vDash pre_e \text{ implies } \mathcal{M}\mathcal{E}, (w, e) \vDash \phi$$

where $\mathcal{M}\mathcal{E} = \langle \Omega \otimes E, \Sigma', (T \times_a P)_{i \in A}, v' \rangle$:
$\Omega \otimes E = \{(w, e) \in \Omega \times E : \mathcal{M}, w \vDash pre_e\}$
$\Sigma' = \sigma\{A \times F : A \in \Sigma, F \in \mathcal{P}(E)\}$

$$T\times_a P((w,e),A,B) = \begin{cases} \dfrac{\sum_{x\in\overline{B}} T_a(w,A_x,\overline{B}_x)\cdot T_a(w,\overline{B}_x,\pi_1(\overline{B}))\cdot P_a(e,\{x\})}{\sum_{x\in E} T_a(w,\overline{B}_x,\pi_1(\overline{B}))\cdot P_a(e,\{x\})} & \text{if } \overline{B}\neq\varnothing \\ 1 & \text{otherwise.}\end{cases}$$

where

$$\overline{B} = \{\,(s,x)\in B : P(e,\{x\})>0\,\},$$
$$\pi_1(\overline{B}) = \{\,s\in W : (s,x)\in\overline{B}\,\}.$$

$v'(p) = \{(w,e) : \mathcal{M}, w \vDash \tau_e(p)\}$.

The definition of $T\times P$ does not seem intuitive on a first look. We prove it is indeed a conditional probability function and provide some intuitions behind its definition on the next section.[1]

It is easy to see that assignments with preconditions are a particular case of event assignment models. To go from the former to the latter one should set the event assignment model to be a singleton with only one event and the probability function P to be equal to 1.

Let's go back to the drawing action as an example for event assignment model.

Example 4.19 (Example 4.16 revisited.)**.** Remember the situation of Algo given by Example 4.16 and Figure 4.3.

Consider the following event model given by $\mathcal{D} = \langle D, pre, P, \tau\rangle$, where

$D = \{d_{c2_3} : c \in \{6B, 7B, 8B, 9B\}\}$

$pre(d_{cp_i}) = \neg(c1 \vee c2) \wedge \bigwedge_{j\leq i, x>c} \neg x2_j \wedge \bigwedge_{i\leq j, c>x} \neg x2_j$

$P(x,\{y\}) = \frac{1}{4}$ for all $x, y \in D$ (Uniform distribution)

$\tau(d_{cp_i}) = \{cp_i := \top\} \cup \{x2_{j+1} := x2_j : x$ is a card with $x \neq c$ and $j \in \{3,4\}\} \cup \{xp_i := \bot : x \neq c\} \cup \{U_{p_{j+1}} := U_{p_j}, D_{p_{j+1}} := D_{p_j} : j \geq i\}$.

Let $\mathcal{M} = \langle \Omega, \Sigma, T, v\rangle$ be the epistemic model after player 1 makes his guesses and before player 2 draws a card; and let w_0 be the current state.

We can express some interesting ideas, for instance, before the drawing, player 1 considers it possible for the card in position 3 in player 2's hand to be $7B$, but after drawing he considers that impossible: $(\neg L^1_1 \neg 7B2_3) \wedge \bigwedge_{d\in D}[\mathcal{D}, d] L^1_1 \neg 7B2_4$.

Formula $(L^1_1 \neg 8B2_3) \wedge \bigwedge_{d\in D}[\mathcal{D}, d] L^1_{.5} 8B2_3$ says that before drawing the card player 1 considers it impossible for player 2 to have $8B$ in that position, but considers it more likely than not after the drawing (perhaps based on player 2's past guesses).

Player 1's beliefs about unrelated cards (like $0W$) are not affected, as expressed by $(L^1_r 0W2_1) \rightarrow \bigwedge_{d\in D}[\mathcal{D}, d] L^1_r 0W2_1$, for any r.

To finish our example lets prove the following fact.

[1] I am thankful to my Advisor, Makoto Kanazawa, for the idea for this function,

Fact: In the model given in Example 4.16 it holds that

$$\mathcal{M}, w_0 \vDash (\neg L_1 \neg 07B2_3) \wedge \bigwedge_{d \in D} [\mathcal{D}, d] L_1 \neg 7B2_4.$$

Proof of the fact: It is easy to see that $\mathcal{M}, w_0 \vDash (\neg L_1^1 \neg 7B2_3)$ holds. For the second part we have that $\mathcal{M}, w_0 \vDash \bigwedge_{d \in D} [\mathcal{D}, d] L_1^1 \neg 7B2_4$ iff $\mathcal{M}, w_0 \vDash pre_d$ implies $\mathcal{M} \times D, (w_0, d) \vDash L_1^1 \neg 7B2_4$ for each $d \in D$. If we prove that $[\![7B2_4]\!]_{\mathcal{M} \times D}$ is empty when $\mathcal{M}, w_0, \vDash pre_d$ we have the result.

Let's prove that $[\![7B2_4]\!]_{\mathcal{M} \times D}$ is the empty set when $\mathcal{M}, w_0, \vDash pre_d$. Suppose that some $(w, e) \in [\![7B2_4]\!]_{\mathcal{M} \times D}$, then $\mathcal{M} \times D, (w, e) \vDash 7B2_4$, which happens only if $(w, e) \in v'(7B2_4) = \{(v, f) : \mathcal{M}, v \vDash \tau_f(7B2_4)\}$, which is equivalent to $\mathcal{M}, w \vDash 7B2_3$. But if $\mathcal{M}, w \vDash 7B2_3$, then $\mathcal{M}, w \nvDash pre_f$ for $f \in D$. Hence $[\![7B2_4]\!]_{\mathcal{M} \times D}$ is empty and the result holds. □

Conditional probability logic is not expressive enough to be able to allow reduction axioms for E-CPL. It is an open question if a Halpern style of conditional probability logic could express a set of reduction axioms for E-CPL.

For the remaining of this chapter we will explain the product between a two conditional probability spaces and the definition of $T \times P$ given earlier. On the next chapter we use *E-CPL* to model actions in Algo.

4.3 Product of Conditional Probability Space.

In this section we define the product between conditional probability measure.

The product between two probability spaces is well understood, see e.g. Halmos (1950). However, the same discussion is more complicated for conditional probability spaces. A more laborious definition of the product seems to be needed.

The literature in probability theory has a definition that serves them well. Let (X, Σ_1, T) and (Y, Σ_2, P) be conditional probability spaces. Let $\Sigma_1 \odot \Sigma_2$ be the smallest σ-algebra containing $\{A_1 \times A_2 : A_1 \in \Sigma_1, A_2 \in \Sigma_2\}$.

Following the definition in Rao (2010), the space resulted of the product between both spaces (X and Y) has its conditional probability measure $T \odot P$ as a function with domain and range seen below.

$$T \odot P : (\Sigma_1 \odot \Sigma_2) \times (\Sigma_1 \times \Sigma_2) \to [0, 1]. \qquad (\dagger)$$

However, the second component of the domain $(\Sigma_1 \times \Sigma_2)$ is a problem for conditional probability logic. It is not difficult to imagine a conditional probability model $\mathcal{M}$, an event model $\mathcal{E}$ and a formula ϕ such that $[\![\phi]\!]_{\mathcal{ME}}$ is not an element of $\Sigma_1 \times \Sigma_2$.[1] The problem arrises when trying to define the satisfaction for a formula of the form $L_r(\top||\phi)$, i.e., what should be $\mathcal{ME}, (w, e) \vDash L_r(\top||\phi)$ if $[\![\phi]\!]$ is not an element in the domain of $T \times P$?

The problem is how to define a product that makes sense for sets (that correspond to the set of states that satisfies a formula) that are outside this particular domain $(\Sigma_1 \times \Sigma_2)$.

Let (W, Σ_1, T) and (E, Σ_2, P) be conditional probability spaces. Denote by Σ the set $(\Sigma_1 \odot \Sigma_2)$, Let $A = A_1 \times A_2$ and $B = B_1 \times B_2$ with $A_1, B_1 \in \Sigma_1$ and $A_2, B_2 \in \Sigma_2$ and define

$$T \odot P(A, B) = T(A_1, B_1,) \cdot P(A_2, B_2).$$

As shown by Rao (2010), if $B \neq \varnothing$, then this equation defines a conditional probability measure over $(\Sigma_1 \odot \Sigma_2) \times (\Sigma_1 \times \Sigma_2)$. It is not difficult to see that in the case $B = \varnothing$ this definition also results in a conditional probability measure. With this idea in mind we know that for our definition of product $T \times P(A, B)$ it should hold that if A and B are rectangles, then we should have$T \times P(A, B) = T \odot P(A, B)$. Next we will see how to extend the function $T \times P$ when B is an union of disjoint rectangles.

Let $B \subset W \times E$ be of the form $B = \bigcup_{i \leq n} B_i$ where for each $i \leq n$ we have $B_i \in \Sigma_1 \times \Sigma_2$, and $B_i \cap B_j = \varnothing$ if $i \neq j$, i.e., B is a finite union of disjoint rectangles. Choose the collection of B_i's such that for each i and j either $\pi_2(B_i) = \pi_2(B_j)$ or $\pi_2(B_i) \cap \pi_2(B_j) = \varnothing$. Define the following subset of B:

$$\bar{B} = \bigcup_{i \in I} B_k,$$

where $I = \{i : P(\pi_2(B_i), \pi_2(B)) > 0\}$. Note that I is always non-empty.

The following lemma is a technical result.

Lemma 4.20. If $A, B \in \Sigma$ are such that $A = \bigcup_{i \leq n} A_i$ and $B = \bigcup_{j \leq m} B_j$ with $A_i \subseteq B_i$ for each $i \leq n$, then $P(A, B) > 0$ implies $\bar{A} \subset \bar{B}$.

[1]Let $\Omega = W \times E$ where $W = \{w_1, w_2\}$ and $E = \{e_1, e_2\}$. Let a valuation v be such that $v(p) = \{(w_1, e_1), (w_1, e_2), (w_2, e_2)\}$. Note that there are no sets A and B such that $A \subset W$, $B \subset E$ and $[\![p]\!] = A \times B$.

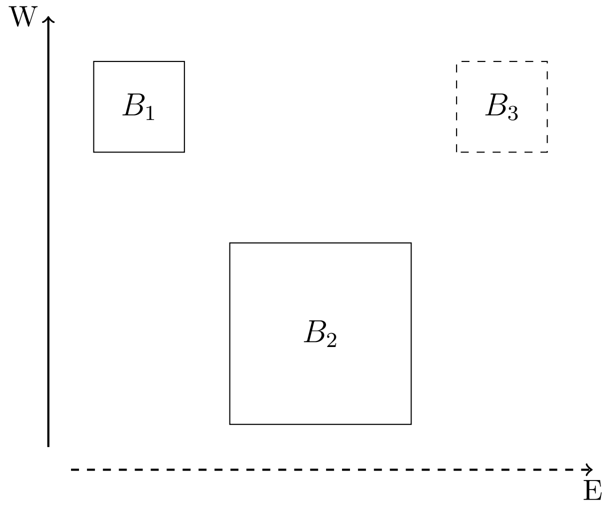

Figure 4.4: In this example $B = B_1 \cup B_2 \cup B_3$. The set $\bar{B}$ is given by $B_1 \cup B_2$.

Proof. Let x be an element of $\bar{A}$, then there is a k such that $x \in A_k$. We know that $P(A_i, A) \cdot P(A, B) = P(A_i, B)$, and since $P(A, B) > 0$ we have that $A_i \subseteq \bar{B}$, hence $x \in \bar{B}$. □

To help simplify notation define λ as the following function:

$$\lambda(B_i, \bar{B}) = T(\pi_1(B_i), \pi_1(\bar{B})) \cdot P(\pi_2(B_i), \pi_2(\bar{B})).$$

Define the product $T \otimes P(B_i, B)$ to be the average of the probabilities among all the B_k given $\bar{B}$ given by the following equation:

$$T \otimes P(B_i, B) = \frac{\lambda(B_i, \bar{B})}{\sum_{k \leq n} \lambda(B_k, \bar{B})} = \frac{T(\pi_1(B_i), \pi_1(\bar{B})) \cdot P(\pi_2(B_i), \pi_2(\bar{B}))}{\sum_{k \leq n} T(\pi_1(B_k), \pi_1(\bar{B})) \cdot P(\pi_2(B_k), \pi_2(\bar{B}))} \quad (4.1)$$

See Figure 4.4 for a representation of $\bar{B}$. Since $\bar{B}$ is not necessary a rectangle we denote this product by $T \otimes P$ to avoid any confusion with the product $T \odot P$ given in Rao (2010) that is defined only over rectangles.

Lemma 4.21. For any $B = B_1 \times B_2 \in \Sigma_1 \times \Sigma_2$ it holds that $T \otimes P(B, B) = 1$.

Proof. For B as in the hypothesis $T \otimes P(B, B)$ is equal to $\dfrac{\lambda(B, \bar{B})}{\lambda(B, \bar{B})}$. The only remaining step is to prove that $\lambda(B, \bar{B}) \neq 0$.

Remember that $\bar{B} \subseteq B$ and consequently $\pi_i(\bar{B}) \subseteq \pi_i(B)$. Therefore $T(\pi_1(B), \pi_1(\bar{B})) = P(\pi_2(B), \pi_2(\bar{B})) = 1$ and we can conclude that $T \otimes P(B, B) = 1$.

□

Denote by $\Sigma = (\Sigma_1 \odot \Sigma_2)$ and by $\Sigma^{\uplus} = \{B : B = \bigcup_{i \leq n} B_i, B_i \in \Sigma_1 \times \Sigma_2$, and $B_i \cap B_j = \varnothing$ if $i \neq j\}$. Each element of $\Sigma^{\uplus}$ is equal to a finite union of pairwise disjoint rectangles. Let $A \in \Sigma_1 \times \Sigma_2$ and $B \in \Sigma^{\uplus}$, then define $T \times P : \Sigma \times \Sigma^{\cup} \to [0, 1]$ as:

$$T \times P(A, B) = \sum_{i \leq n} (T \odot P)(A, B_i) \cdot (T \otimes P)(B_i, B) \tag{4.2}$$

The intuition behind this equation is the fact that if B_1 and B_2 are disjoint, then for any conditional probability function μ the probability of A given $B_1 \cup B_2$ is given by

$$\mu(A, B_1 \cup B_1) = \mu(A, B_1)\mu(B_1, B_1 \cup B_2) + \mu(A, B_2)\mu(B_2, B_1 \cup B_1).$$

This equation relates that the probability of a set given a union of two disjoint sets is the sum of each probability given the sets separated and weighted by its own relative probabilities.

Substituting Equation 4.1 in Equation 4.2 we have the following:

$$T \times P(A, B) = \frac{\sum_{i \leq n} T \odot P(A, B_i) \cdot T(\pi_1(B_i), \pi_1(\bar{B})) \cdot P(\pi_2(B_i), \pi_2(\bar{B}))}{\sum_{k \leq n} T(\pi_1(B_k), \pi_1(\bar{B})) \cdot P(\pi_2(B_k), \pi_2(\bar{B}))}$$

To simplify our notation from now on we write A^i for $\pi_i(A)$.

Remember that $T \odot P(A, B_i) = T(A^1, B_i^1) \cdot P(A^2, B_i^2)$. Substituting that on the previous equation we have:

$$T \times P(A, B) = \frac{\sum_{i \leq n} T(A^1, B_i^1) \cdot P(A^2, B_i^2) \cdot T(B_i^1, \bar{B}^1) \cdot P(B_i^2, \bar{B}^2)}{\sum_{k \leq n} T(B_k^1, \bar{B}^1) \cdot P(B_k^2, \bar{B}^2)} \tag{4.3}$$

Figure 4.5 illustrates a possible relation between A, B and B^*.

We summarize the previous discussion with the following theorem. As the objective of this text is for event models after we state and prove the existence of product conditional probabilities for the case where the second coordinate is finite.

Theorem 4.22. Let (W, Σ_1, T) and (E, Σ_2, P) be conditional probability spaces. Let Σ be the smallest σ-algebra in $W \times E$ containing $\Sigma_1 \times \Sigma_2$. Let $\Sigma^{\uplus}$ be the set of all finite

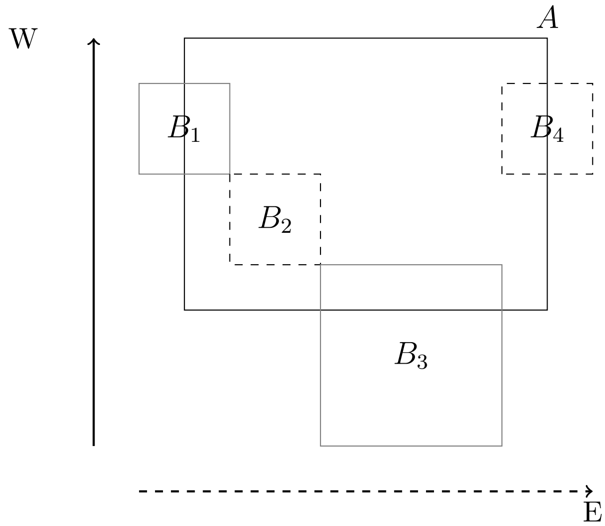

Figure 4.5: In this example $\bar{B} = B_1 \cup B_3$. The probability $T \times P(A, B)$ is equal to $\sum_{i \leq 4} T \odot P(A, B_i) \cdot T \otimes P(B_i, B)$.

disjoint union of elements of $\Sigma_1 \times \Sigma_2$, i.e., $\Sigma^{\uplus} = \{B : B = \bigcup_{i \leq n} B_i, B_i \in \Sigma_1 \times \Sigma_2$, and $B_i \cap B_j = \varnothing$ if $i \neq j\}$.

Then there is a conditional probabilistic measure, which we denote by $T \times P$, on the product space $(W \times E, \Sigma, \Sigma^{\uplus})$ such that if $A_1, B_1 \in \Sigma_1$ and $A_2, B_2 \in \Sigma_2$, then the following holds:

$$T \times P(A_1 \times A_2, B_1 \times B_2) = T(A_1, B_1) \cdot P(A_2, B_2).$$

Moreover, for any $A \in \Sigma$ and $B \in \Sigma^{\uplus}$ the following equation defines one possible $T \times P$:

$$T \times P(A, B) = \frac{\sum_{i \leq n} T(A^1, B_i^1) \cdot P(A^2, B_i^2) \cdot T(B_i^1, \bar{B}^1) \cdot P(B_i^2, \bar{B}^2)}{\sum_{k \leq n} T(B_k^1, \bar{B}^1) \cdot P(B_k^2, \bar{B}^2)} \tag{4.4}$$

Where $\pi_i(A) = A^i$ and $\bar{B} = \bigcup_{i \in I} B_k$ with $I = \{i : P(\pi_2(B_i), \pi_2(B)) > 0\}$.

Proof. First let A, B be elements in Σ such that there are A_i, B_i in Σ_i for $i \in \{1, 2\}$ and $A = A_1 \times A_2$ and $B = B_1 \times B_2$. A and B are rectangles.

Let's show that $T \times P(A, B) = T(A_1, B_1) \cdot P(A_2, B_2)$.

By Equation 4.2 we have $T \times P(A, B) = T \odot P(A, B) \cdot T \otimes P(B, B)$. By Lemma 4.21 we know that $T \times P(A, B) = T \odot P(A, B)$. And as was already shown by Rao (2010), if

$B \neq \varnothing$ it holds that $T \odot P(A, B) = T(A_1, B_1) \cdot P(A_2, B_2)$. With that we can easily see that $T \times P(A, B) = T(A_1, B_1) \cdot P(A_2, B_2)$.

Now let's show that $T \times P$ as given by Equation 4.4 is indeed a conditional probability measure.

(i) If $A \in \Sigma^{\uplus}$, then $T \times P(A, A) = 1$.

This is straightforward as seen in the following:

$$T \times P(A, A) = \frac{\sum_{i \leq n} T \odot P(A, A) \cdot \lambda(A_i, \bar{A})}{\sum_{i \leq n} \lambda(A_i, \bar{A})} = \frac{\sum_{i \leq n} \lambda(A_i, \bar{A})}{\sum_{i \leq n} \lambda(A_i, \bar{A})} = 1$$

(ii) $T \times P(\cdot, B)$ is σ-additive.

Let $(A_j)_{j \in J}$ be a countable pairwise disjoint sequence of elements of Σ, then:

$$T \times P(\bigcup_{j \in J} A_j, B) = \sum_{i \leq n} T \odot P(\bigcup_{j \in J} A_j, B_i) \cdot T \otimes P(B_i, B) =$$

$$= \sum_{i \leq n} (\sum_{j=0}^{\infty} T \odot P(A_j, B_i)) \cdot T \otimes P(B_i, B) = \sum_{j=0}^{\infty} (\sum_{i \leq n} T \odot P(A_j, B_i) \cdot T \otimes P(B_i, B)) =$$

$$= \sum_{j=0}^{\infty} T \times P(A_j, B).$$

(iii) $T \times P(A \cap B, C) = T \times P(A, B \cap C) \cdot T \times P(B, C)$.

We use an equivalent definition of conditional probability measure given by Rao (2010). For any $A \in \Sigma$ and $B, C \in \Sigma^{\uplus}$ we have to prove that:

(1) $T \times P(A \cap B, B) = T \times P(A, B)$; and

(2) if $A \subseteq B \subseteq C$, then $T \times P(A, B) \cdot T \times P(B, C) = T \times P(A, C)$.

Property (1) holds because of the properties of $T \odot P$. We now prove property (2).

Let $A \in \Sigma$ and $B, C \in \Sigma^{\uplus}$ be such that $A \subseteq B \subseteq C$. First assume that A, B and C are non-empty.

Let $(B_i)_{i \leq n}$ and $(C_j)_{j \leq m}$ be sequences of pairwise disjoint elements of $\Sigma_1 \times \Sigma_2$ such that $B = \bigcup_{i \leq n} B_i$ and $C = \bigcup_{j \leq m} C_j$. Moreover choose the sequence $(C_j)_{j \leq m}$ such that $n \leq m$, $C_j = B_j$ if $j \leq n$ and $C_j \cap B = \varnothing$ if $n < j \leq m$. It is always possible to construct such sequence of C_i's.

Using the fact that T and P are both conditional probability measures we can write $T \times P(A, B) \cdot T \times P(B, C)$ as follows:

$$\frac{\sum_{i \leq n} T(A^1 \cap B_i^1, \bar{B}^1) \cdot P(A^2 \cap B_i^2, \bar{B}^2)}{\sum_{k \leq n} T(B_k^1, \bar{B}^1) \cdot P(B_k^2, \bar{B}^2)} \cdot \frac{\sum_{i \leq m} T(B^1 \cap C_i^1, \bar{C}^1) \cdot P(B^2 \cap C_i^2, \bar{C}^2)}{\sum_{k \leq m} T(C_k^1, \bar{C}^1) \cdot P(C_k^2, \bar{C}^2)} \quad (4.5)$$

First note that in the case that $T(\bar{B}^1, \bar{C}^1) \cdot P(\bar{B}^2, \bar{C}^2) = 0$ then one of the factors is also zero. If $T(\bar{B}^1, \bar{C}^1) = 0$ we must have that $T(B^1 \cap C_i^1, \bar{C}^1) = 0$ for all $i \leq m$, hence $T \times P(B, C) = 0$. On the other hand, since $A \subseteq B$, it also must hold that $T(\bar{A}^1, \bar{C}^1) = 0$, therefore the equation $T \times P(A, B) \cdot T \times P(B, C) = T \times P(A, C)$ holds. The argument is similar in the case that $P(\bar{B}^2, \bar{C}^2) = 0$.

In the case that $T(\bar{B}^1, \bar{C}^1) \cdot P(\bar{B}^2, \bar{C}^2) \neq 0$ we can multiply the left factor in Equation 4.5 by

$$\frac{T(\bar{B}^1, \bar{C}^1) \cdot P(\bar{B}^2, \bar{C}^2)}{T(\bar{B}^1, \bar{C}^1) \cdot P(\bar{B}^2, \bar{C}^2)}$$

This together with the facts that T and P are conditional probability measures and since $\bar{B} \subseteq \bar{C}$ (by Lemma 4.20) result in the following:

$$\frac{\sum_{i \leq n} T(A^1 \cap B_i^1, \bar{C}^1) \cdot P(A^2 \cap B_i^2, \bar{C}^2)}{\sum_{k \leq n} T(B_k^1 \cap \bar{B}^1, \bar{C}^1) \cdot P(B_k^2 \cap \bar{B}^2, \bar{C}^2)} \cdot \frac{\sum_{i \leq m} T(B^1 \cap C_i^1, \bar{C}^1) \cdot P(B^2 \cap C_i^2, \bar{C}^2)}{\sum_{k \leq m} T(C_k^1, \bar{C}^1) \cdot P(C_k^2, \bar{C}^2)}$$

Because of the way we chose each C_i we have that $T(B^1 \cap C_i^1, \bar{C}^1) \cdot P(B^2 \cap C_i^2, \bar{C}^2) = 0$ for all $i > n$ and for each $i \leq n$ it holds that $T(B^1 \cap C_i^1, \bar{C}^1) \cdot P(B^2 \cap C_i^2, \bar{C}^2) = T(B_i^1, \bar{C}^1) \cdot P(B_i^2, \bar{C}^2)$. So, we have the last equation equal to the following:

$$\frac{\sum_{i \leq n} T(A^1 \cap B_i^1, \bar{C}^1) \cdot P(A^2 \cap B_i^2, \bar{C}^2)}{\sum_{k \leq n} T(B_k^1 \cap \bar{B}^1, \bar{C}^1) \cdot P(B_k^2 \cap \bar{B}^2, \bar{C}^2)} \cdot \frac{\sum_{i \leq n} T(B_i^1, \bar{C}^1) \cdot P(B_i^2, \bar{C}^2)}{\sum_{k \leq m} T(C_k^1, \bar{C}^1) \cdot P(C_k^2, \bar{C}^2)}$$

For each i such that $B_i \cap \bar{B} \neq B_i$, then $P(B_k, \bar{C}) = 0$ and the whole term $T(B_i^1, \bar{C}^1) \cdot P(B_i^2, \bar{C}^2)$ is equal to zero, hence we can cancel out the denominator from the left factor with the numerator from the right fact in last equation, resulting in:

$$T \times P(A, B) \cdot T \times P(B, C) = \frac{\sum_{i \leq n} T(A^1 \cap B_i^1, \bar{C}^1) \cdot P(A^2 \cap B_i^2, \bar{C}^2)}{\sum_{k \leq m} T(C_k^1, \bar{C}^1) \cdot P(C_k^2, \bar{C}^2)}$$

To conclude that this equation is equal to $T \times P(A, C)$ note that, since $A \subseteq B$ it holds that

$$\sum_{n < i \leq m} T(A^1 \cap C_i^1, \bar{C}^1) \cdot P(A^2 \cap C_i^2, \bar{C}^2) = 0.$$

□

Lemma 4.23. *The definition of $T \times P$ is well-defined, i.e., it is independent of the choice of rectangles that cover a set.*

Corollary 4.24. With the same hypothesis as before, if (E, Σ_2) is also finite discrete we can calculate $T \times P$ as follows:

Since E is finite we can rewrite the definition of $T \times P$ as follows.

$$T \times P(A, B) = \frac{\sum_{x \in E} T(A_x, \bar{B}_x) \cdot T(\bar{B}_x, \bar{B}^1) \cdot P(\{x\}, \bar{B}^2)}{\sum_{x \in E} T(\bar{B}_x, \bar{B}^1) \cdot P(\{x\}, \bar{B}^2)}$$

Chapter 5

Algo

Repara bem no que não digo.

Paulo Leminski

Listen carefully to what I ain't saying.

Paulo Leminski

5.1 Introduction

The computer scientists are often interested in automated players. For instance, Billings et al. (2002) discussed an automated player for poker. On the other hand, van Ditmarsch (2001) with a non-probabilistic language modeled a game called *Cluedo* in terms of the players' knowledge states and how they change after public announcements are made and private information is shared.

Algo lacks formal studies so in this chapter, as an application of conditional probabilistic language we express strategies of card games using Algo as example. The game allows up to 4 players; however we will restrict our analysis for the 2 player case.

5.2 Defining the game formally

Let N be the set of natural numbers less than n, i.e., $N = \{m \in \mathbb{N} : m < n\}$. Let I be a set of symbols for color. On the standard *Algo* $I = \{b, w\}$ where b and w stand for black and white. A *card* is an element of $\mathbb{C} = N \times I$. If c is a card we denote its value and color by $value(c)$ and $color(c)$. Define the order $<$ over $\mathbb{C}$ as the lexicographic order, i.e., $vc < v'c'$ iff $v < v'$ or $v = v'$ and $c < c'$, where $b < w$.

Definition 5.1. An *Algo state* s is a tuple $s = \langle h_1, h_2, f\rangle$ where h_1 and h_2 are the sequences of cards held by player 1 and 2. We denote by $h_1(i)$ and $h_2(j)$ the i-th and j-th card in the sequences. We require $h_1(i) < h_1(j)$ if $i < j$; furthermore we ask for $h_1(i) \neq h_2(j)$ for any i, j. If $H = \{c : c = h_1(i) \text{ or } c = h_2(j) \text{ for some } i \text{ or } j\}$ is the set of cards in the hands of the players, let f be a function $f : H \to \{0, 1\}$ which says if a card is face up or down ($f(c) = 0$ if c is face down and $f(c) = 1$ if c is face up).

We call *universe* the set U of all Algo states. If $s = \langle h_1, h_2, f\rangle$ is an Algo state we denote by $hand_i(s)$ the sequence h_i. Often we will denote the sequence h_1 and h_2 as sets; we believe no confusion will rise.

We assume a random function *pile* which receives an Algo state and returns a card that neither player holds, if any. Formally:

$pile(s) \in \mathbb{C} \setminus (hand_1(s) \cup hand_2(s)$ if $\mathbb{C} \setminus (hand_1(s) \cup hand_2(s) \neq \varnothing$,

$pile(s) = \varnothing$ otherwise.

From now on we shall omit the index n, but the reader should keep in mind that all the definitions should change if the number of cards in the game were to change.

Definition 5.2 (Similar hands:$\leftrightarrow_i$)**.** We say that two Algo states $s = \langle h_1, h_2, f\rangle$ and $s' = \langle h'_1, h'_2, f'\rangle$ are *similar for player i* if for $j \neq i$ we have that:

1. $h_i = h'_i$ and $f(h_i(k)) = f'(h_i(k))$ for all $k \leq |h_i|$
2. h_j and h'_j have the same size;
3. for every $k \leq |h_j|$ we have $color(h_j(k)) = color(h'_j(k))$;
4. for every $k \leq |h_j|$ we have $f(h_j(k)) = f'(h'_j(k))$ and if $f(h_j(k)) = 1$, then $h_j(k) = h'_j(k)$.

Let X be a set and $<$ a linear order over X. Let $Y \subset X$. We denote by $sort(Y)$ the sequence $(y_1, y_2, ...)$ such that $Y = \{y_1, y_2, ...\}$ and $y_i < y_j$ if $i < j$.

Definition 5.3. A *draw* D_1 is a relation over U defined as follows: for $s = \langle h_1, h_2, f\rangle$ and $s' = \langle h'_1, h'_2, f'\rangle$ elements of U, sD_1s' iff $h_2 = h_2$, $h'_1 = sort(pile(s) \cup h_1)$, and $f'(x) = f(c)$ if $c \neq pile(s)$, $f'(pile(s)) = 0$. Similarly we define D_2. We denote by D the sequence (D_1, D_2).

An *epistemic structure* for Algo is a tuple $\mathcal{M} = \langle W, (\leftrightarrow_i)_{i\leq 2}, s\rangle$ where $W \subset U$, $\leftrightarrow_i$ is defined as follows: For every $s, s' \in W$ we say that $s \leftrightarrow_i s'$ if $h_i = h'_i$ and s and s' are similar for player i; s is an Algo state.

Remember that if R is a relation, R^* denotes the reflexive transitive closure of R.

An *initial* epistemic structure is a tuple $\mathcal{M} = \langle W_0, (\leftrightarrow_i)_{i\leq 2}, s_0\rangle$, where s_0 is an Algo state with the constraints $|hand_i(s_0)| = 4$, $f(x) = 0$ for every x a card. $W_0 = \{s \in U : s_0(\leftrightarrow_1 \cup \leftrightarrow_2)^* s\}$.

5.3 Language

In this section we define a language for Algo and express the strategies in this formal language. We start showing the language, then we build the models to interpret the language of Algo.

We restate our set of propositions as mentioned in the introduction. Let the set of propositions be $P_c = \{vcp_i : v < n, c \in \{B, W\}$, p is a player, $i \leq n\}$, v stands for the value of the card, c stands for the color, p stands for the player holding the color and i stands for the position in the hand of the player. Fix also the following set of propositions $P_f = \{U_{p_i}, D_{p_i} : i \leq n$, p is a player$\}$, where U_{p_i} stands for 'card in position i of player p is face up' and D_{p_i} stands for 'card in position i of player p is face down.' Note that $(\neg U_{p_i}) \wedge (\neg D_{p_i})$ is not a contradiction, it implies that i exceed the size of player p's hand. The set of Propositions is $P = P_c \cup P_f$.

The set of players of Algo is denoted by A and we call an element a of A a player. We assume that A has only two elements and sometimes refers as player 1 and player 2 each of the elements of A.

The language for Algo is given by:

$$\begin{aligned} \phi &\doteq\ p \mid \neg\phi \mid \phi \wedge \phi \mid L^a_r(\phi||\phi) \mid \Box\phi \mid \alpha \\ \alpha &\doteq\ [D, d]\phi \mid [F, f]\phi \mid [!g]\phi \mid [!!g]\phi \end{aligned}$$

Where $p \in P$ and $a \in A$. D and F are the event-assignment for drawing a card and for turning a card face up, which are defined below. Finally, g is a guess and the operators $[!g]$ and $[!!g]$ are also defined below. α denotes the action part of the language, and is the focus of this section.

Figure 5.1 presents the axioms for Algo$_n$. Axiom (Al_1) says that a card cannot be face up and face down at the same time. Axiom (Al_2) says that a card cannot be in

· (Al_1) $\neg(D_{a_i} \wedge U_{a_i})$, for all $i \leq n$ and all players a

· (Al_2) $\neg(vca_i \wedge vcb_j)$ if $a \neq b$ or $i \neq j$

· (Al_3) $\neg(vca_i \wedge v'c'a_j)$ if $i \leq j$ and $vc > v'c'$

· (Al_4) $(\bigvee_{v,c} vca_i) \leftrightarrow (D_{a_i} \vee U_{a_i})$, for all $i \leq n$

Figure 5.1: Axioms of Algo$_n$

two positions or with two different players at the same time. Axiom (Al_3) says that the hand should be ordered. Finally, axioms (Al_4) states the idea that if there is a card in position i then it is face up or face down.

Notation To help us with the notation we make the following abbreviation for any value $v \leq n$ and color $c \in \{B, W\}$:

· $vcp \doteq \bigvee_{j \leq n} vcp_j$

· $vc \doteq \bigvee_{i \in Player} vci$

In this way a formula vcp stands for the fact that the card vc is in player p's hand. And the formula vc stands for the fact that the card vc is in the hand of one of the players.

Static Interpretation

Let $\mathcal{M} = \langle W, (\leftrightarrow_a)_{a \in A}, s_0 \rangle$ be an epistemic structure for Algo, we define a *CP-model for Algo* as $\mathcal{M}_{cp} = \langle W, \Sigma, (T_a)_{a \in A}, v \rangle$ where $\Sigma = \mathcal{P}(W)$. Let $\bar{s}^a = \{t \in W : s \leftrightarrow_a t\}$, then we define T_a :

$$T_a(s, A, B) = \begin{cases} \frac{|A \cap B \cap \bar{s}^a|}{|B \cap \bar{s}^a|} & \text{if } B \cap \bar{s}^a \neq \varnothing \\ \frac{|A \cap B|}{|B|} & \text{if } B \cap \bar{s}^a = \varnothing \text{ and } B \neq \varnothing \\ 1 & \text{otherwise.} \end{cases} \tag{5.1}$$

The valuation v is defined as:

$$v(p) = \begin{cases} \{s \in W : hand_p(s)(n) = vc\} & \text{if } p = vcp_n \\ \{s \in W : f(hand_p(s)(i)) = 1\} & \text{if } p = U_{p_i} \\ \{s \in W : f(hand_p(s)(i)) = 0\} & \text{if } p = D_{p_i} \end{cases}$$

$T_a(s, A, B)$ is the probability that agent a with the information in state s assigns to the fact that the actual state (possibly different than s) is in A given it is in set B.

The idea behind the second clause of the definition of T_a is that if the information in the state is conflicted with the new information the old information should be discarded. The choice of T_a is not unique. One could make the case for different choices with different policies for the empty set caused by the contradiction.

Let us prove that T_a as defined is a Conditional Markov Kernel.

Theorem 5.4. The function T_a as defined above is a conditional Markov kernel.

Proof. Fix $s \in W$ and $A, B, C \subseteq W$. For simplicity abbreviate $T_a(s, A, B)$ as $T(A, B)$ and $\bar{s}^a$ by s.

(Condition 1) $T(A, B) \geq 0$; $T(B, B) = 1$.

To prove that $T(A, B) \geq 0$ it is enough to note that $\frac{|A \cap B \cap s|}{|B \cap s|}$, $\frac{|A \cap s|}{|s|}$ are always greater or equal to zero.

To prove that $T(B, B) = 1$ note that $\frac{|B \cap s|}{|B \cap s|} = \frac{|B|}{|B|} = 1$.

(Condition 2) For any $B \neq \varnothing$ holds that $T(\cdot, B)$ is σ-additive.

Since W is finite it is enough to prove additivity. Let A_1 and A_2 be disjoint subsets of W. If $B \cap s \neq \varnothing$ we have that

$$T(A_1 \cup A_2, B) = \frac{|(A_1 \cup A_2) \cap (B \cap s)|}{|B \cap s|} = \frac{|(A_1 \cap B \cap s) \cup (A_2 \cap B \cap s)|}{|B \cap s|},$$

and since A_1 and A_2 are disjoint it holds that the right-hand side of the last equality is equal to the led-hand side of the next equality:

$$\frac{|(A_1 \cap B \cap s)|}{|B \cap s|} + \frac{|A_2 \cap B \cap s|}{|B \cap s|} = T(A_1, B) + T(A_2, B)$$

If $B \cap s = \varnothing$ the reasoning is similar.

(Condition 3) $T(A \cap B, C) = T(A, B \cap C) \cdot T(B, C)$.

If $C \cap s$ and $B \cap C \cap s$ are nonempty the conclusion is straightforward.

If $C \cap s \neq \varnothing$ and $B \cap C \cap s = \varnothing$, then $T(A \cap B, C) = T(B, C) = 0$. Therefore $T(A \cap B, C) = T(A, B \cap C) \cdot T(B, C)$.

If $C \cap s = \varnothing$, then $B \cap C \cap s = \varnothing$, and the conclusion is straightforward.

If $C = \varnothing$, then $B \cap C = \varnothing$ and $T(A \cap B, C) = T(A, B \cap C) = T(B, C) = 1$, hence the conclusion.

Therefore T_a is a conditional Markov kernel. □

We turn now to the definition of truth.

Definition 5.5. We define satisfaction as follows:
$\mathcal{M}_{cp}, s \vDash (p)$ iff $s \in v(p)$;
$\mathcal{M}_{cp}, s \vDash \neg\phi$ iff $\mathcal{M}, s \nvDash \phi$;
$\mathcal{M}_{cp}, s \vDash \phi \wedge \psi$ iff $\mathcal{M}, s \vDash \phi$ and $\mathcal{M}, s \vDash \psi$;
$\mathcal{M}_{cp}, s \vDash L^a_r(\phi||\psi)$ iff $T_a(s, [\![\phi]\!], [\![\psi]\!]) \geq r$

We will mostly denote a card vc by c and make its value explicit only when necessary. Next we will define the event assignment models for the actions in Algo.

5.3.1 Drawing a card

Drawing a card is an event that changes the beliefs of the players and the state of the game, hence we define an event assignment model for this action.

Definition 5.6. Let's define the event model for player a drawing a card. Denote by b his/her opponent and let $\mathcal{D}_a = \langle D, pre, (P_i)_{i\in A}, \tau\rangle$, where:
$D = \{d_{c_i} : c \in \mathbb{C}, i \leq n\}$,

$$pre(d_{c_i}) = \neg(c_a \vee c_b) \wedge \bigwedge_{j<i, x>c} \neg x_{a_j} \wedge \bigwedge_{j\geq i, c>x} \neg x_{a_j},$$

$$P_a(d, d') = \begin{cases} 1 & \text{if } d = d' \\ 0 & \text{if otherwise.} \end{cases}$$

$$P_b(d, d') = \begin{cases} 0 & \text{if } d \not\sim_b d' \\ 1/|\bar{d}^b| & \text{if otherwise.} \end{cases}$$

where $d_{vc_i} \sim_b d_{v'c'_k}$ iff it holds that $c = c'$, $i = k$ and ($a = b$ implies $v = v'$). Also, $\bar{d}^b = \{d' : d \sim_b d'\}$.

$\tau(d_{c_i}) = \{ca_i := \top\} \cup \{xa_{j+1} := xa_j : x$ is a card with $x \neq c$ and $i < j \leq |hand_a|\} \cup \{xa_i := \bot : x \neq c\} \cup \{U_{a_{j+1}} := U_{a_j}, D_{a_{j+1}} := D_{a_j} : j \geq i\}$.

The precondition of drawing a card c that is placed in position i (pre(d_{c_i})) says that neither player holds the card; that any card to the left of the card is smaller and any card to the right is greater than the card c.

The assignment $\tau(d_{c_i})$ says that the card c is face down and in position i in the hand of player a. Also, that any card to the right moves one position to the right (card in position i becomes the card in position $i + 1$, for instance). Remember that in the definition of $\tau(d_{c_i})$ we denote by xa_i the card $x \in \mathbb{C}$ in position j in the hand of player a.

The previous definition is the general case of Example 4.16. We define the meaning of our connective $[\mathcal{D}_a, d]$ through event-assignment model, i.e.,

$$\mathcal{M}_{cp}, s \vDash [\mathcal{D}_a, d]\phi \text{ iff } \mathcal{M}_{cp}, s \vDash pre_d \implies \mathcal{M}_{cp} \times \mathcal{D}_a, (s, d) \vDash \phi$$

Let's illustrate with an example:

Example 5.7. If player 1 draws the card $2B$ in position 2, player 2 considers it impossible for it to be a $3W$. With the formal language we have $[\mathcal{D}_1, d_{2B_2}]M_0^2(3W1_2)$. Let's prove that this formula holds in any model.

Let $\mathcal{M}$ be an epistemic structure for Algo and $\mathcal{M}_{cp}$ its CP-model. For any Algo state s, note the following:

$\mathcal{M}_{cp}, s \vDash [\mathcal{D}_1, d_{2B_2}]M_0^2(3W1_2)$ which is equivalent to

$\mathcal{M}_{cp}, s \vDash pre(d_{2B_2})$ implies $\mathcal{M}_{cp} \times \mathcal{D}, (s, d_{2B_2}) \vDash M_0^2(3W1_2)$ if and only if $\mathcal{M}_{cp}, s \vDash pre(d_{2B_2})$ implies $T \times_2 P(s, d_{2B_2}) = 0$.

The last sentence is true when s does not satisfy the preconditions of d_{2B_2} or, in the case it does and the product is equal to zero. Note that for any $d \in \{d_{iW_j} \in D : i < n, j \leq |hand_1|\}$ we have $P_2(d_{2B_2}, d) = 0$ because $d_{2B_2} \not\sim_2 d$. On the other hand if $d \in \{d_{iB_j} \in D : i < n, j \leq |hand_a|\}$ we have that $[\![3W1_2]\!]_d = \{t : \mathcal{M}_{cp} \times \mathcal{D}_1, (t, d) \vDash 3W1_2\} = \varnothing$, and since $\Omega_d \neq \varnothing$, $T_2(s, [\![3W1_2]\!]_d, \Omega_d) = 0$ for any d. Hence, $T \times_2 P(s, d_{2B_2}, [\![3W1_2]\!], [\![\top]\!]) =$

$$\frac{\sum_{x \in \overline{[\![\top]\!]}} T_2(w, [\![3W1_2]\!]_x, \overline{[\![\top]\!]}_x) \cdot T_2(w, \overline{[\![\top]\!]}_x, \pi_1(\overline{[\![\top]\!]})) \cdot P_2(d_{2B_2}, \{x\})}{\sum_{x \in E} T_2(w, \overline{[\![\top]\!]}_x, \pi_1(\overline{[\![\top]\!]})) \cdot P_2(d_{2B_2}, \{x\})} = 0,$$

holds.

Therefore the formula $[\mathcal{D}_1, d_{2B_2}]M_0^2(3W1_2)$ is always true.

5.3.2 Turning a Card Face Up

From a theoretical point of view turning a card face up should be equivalent to an announcement of the value of the card. However, the action changes the state of the world and we will follow this idea. Turning a card face up is given by an assignment-precondition event which we define as follows:

Definition 5.8. The event model for turning a card face up is the tuple $\mathcal{F} = \langle F, pre_f, P, \tau_f \rangle$ where:

$F = \{f_{ca_i} : c \in \mathbb{C}, a \in A, i \leq n\}$

$pre(f_{a_i}) = ca_i \wedge D_{a_i}$

$\tau(f_{a_i}) = [D_{a_i} := U_{a_i}]$

Different to the connective for drawing a card we define the meaning of the connective $[f]$ through assignment-precondition connective:

$$\mathcal{M}_{cp}, s \models [f]\phi \text{ iff } \mathcal{M}_{cp}, s \models pre(f) \implies (\mathcal{M}_{cp})_f, s \models \phi$$

5.3.3 Guesses

The announcement of a guess is one of the biggest challenges in modeling the game Algo. A definitive modeling and effective strategy is beyond the scope of this thesis, as it relies on game theoretic concepts. In the following we present some interesting points and issues when treating the attack action of a game.

First, we assume that the strategy applied by the attacker on a guess is known by the defender (in fact it is common knowledge). This makes the announcement of a guess a public announcement of a formula.

Naive Guess

After a guess the defender eliminates the possibility that the attacker holds the value of the guessed card.

Let g be the guess $Pa \xrightarrow{cd_j} Pd$, we define the connective $[!g]$ to be the announcement of guess g. Let $\psi_g \doteq \neg(\bigvee_{i \leq |hand_a|} ca_i)$. We interpret $[!g]$ as follows:

$$\mathcal{M}_{cp}, s \models [!g]\phi \text{ iff } \mathcal{M}_{cp}, s \models \psi_g \Longrightarrow (\mathcal{M}_{cp})_{\psi_g}, s \models \phi$$

Remember that $(\mathcal{M}_{cp})_{\psi_g}$ is the updated model after the public announcement of ψ_g, the intended meaning of this formula is 'player a (the attacker) does not hold card c.' Since the operator $[!g]$ is interpreted as the announcement of ψ_g we could have defined the operator $[!g]$ as $[!\phi_g]$, we choose to make clear the idea of announcing a guess g.

Essentially, the announcement of a guess under the assumption that the player is naive is rephrased with the public announcement of the player not holding the guessed card.

Clever Guess

When a player guesses a card which he/she believes to be the most likely (i.e., most probable) we say that he/she is employing the clever strategy. As we saw in the previous paragraph, the essential part of defining a strategy (that is common knowledge among the players) is to find a formula in the language to express the corresponding announcement. Since maximality is a concept outside of our language, we will define a formula to express maximality with finite accuracy.

Fix an epistemic structure $\mathcal{M}$ for Algo and let $\mathcal{M}_{cp}$ be its CP-model. Note that in the state s a card c is the most likely (i.e., most probable) for player a to be in position i on the hand of player d iff for all $r \in [0,1]$ and for all cards x, it holds that $\mathcal{M}_{cp}, s \vDash L^a_r x d_i \to L^a_r c d_i$.[1]

Hence, a guess g given by $Pa \xrightarrow{cd_j} Pd$ is (one of) the most likely guess to be correct for player a iff for all $r \in [0,1]$, for all $i \leq n$ and for all cards x holds that $\mathcal{M}_{cp}, s \vDash L^a_r x d_i \to L^a_r c d_j$.

A limitation of the language should be clear now. To express maximality one needs an infinite disjunction of the form $\bigwedge_{r,i,x} L^a_r x d_i \to L^a_r c d_j$, which is clearly outside our starting language. However, if the set of indexes r is finite we can rewrite the formula with finite conjunctions, instead.

Let $I_k = \{\frac{j}{k} : 0 \leq j \leq k\}$ and define the following formula:

$$\psi_{g^+} \doteq \bigwedge_{r \in I_k, i \leq n, x \in Card} (L^a_r x d_i \to L^a_r c d_j)$$

Although the formula ψ_{g^+} is not equivalent to maximality, the bigger the accuracy of I_k (i.e., the larger the k) the closer it gets. Note that it is enough to bound k by $|\{s' : s' \leftrightarrow_a s\}|$, since this set is bounded by some constant in function of the number of cards.

In later states of the game k is often small (close to two or three). One could also fix a k_0 and have the idea of *good enough*, instead.

We define the connective $[!!g]$ to be the announcement of guess g (for the clever guess). We interpret $[!!g]$ as follows:

$$\mathcal{M}_{cp}, s \vDash [!!g]\phi \text{ iff } \mathcal{M}_{cp}, s \vDash \psi_{g^+} \Longrightarrow (\mathcal{M}_{cp})_{\psi_{g^+}}, s \vDash \phi$$

[1]This holds because of the following: If for all a, $y \geq a$ implies $x \geq a$, then $x \geq y$.

5.4 Player Types

To finish this chapter we define the notion of player types using formulas from the language of Algo defined in the previous section.

A player in Algo has three moments to perform an action/make a choice:

1. The player must choose a guess to attack;
2. If the guess is correct he/she must decide whether to keep attacking or stop; and
3. When attacked he/she should decide how to interpret the attack.

If player 1 chooses a guess on the criteria that he does not hold the guessed card we say that:

On the Algo state s player 1 announces any guess $g = P1 \xrightarrow{c2_{pos}} P2$ that satisfies

$$\mathcal{M}_{cp}, s \models \neg U_{2_{pos}} \wedge \neg c1;$$

remember that the proposition $\neg U_{2_{pos}}$ says that the card in position *pos* in the hand of player 2 is face down. The formula $\neg c1$ stands for 'player 1 does not have the card c'

If player 1 chooses to keep attacking only if his probability of guessing is greater than k_0 we say that:

Player 1 keeps attacking in the state s if $g = P1 \xrightarrow{c2_{pos}} P2$ is his/her next guess and holds

$$\mathcal{M}_{cp}, s \models L^1_{k_0} c2_{pos},$$

where k_0 is the maximum risk Player 1 is willing to take, if the chance of guessing correctly is lower than k_0 he will stop guessing and pass his turn.

Finally, if guess g was to be announced, the formula ψ_g has two roles. First the model should be updated accordingly. This means that if $\mathcal{M}$ is the old model, the new model has as support $[\![\psi_g]\!]_{\mathcal{M}}$. This update happens before the confirmation of the guess from the opponent or before any card turns face up.

These three formulas $(\neg U_{2_{pos}} \wedge \neg c1, L^1_{k_0} c2_{pos}, \psi_g)$ fully describe how player 1 should play the game. We call that the type of player 1.

Definition 5.9. The *player type* is given by a tuple (f_1, f_2, f_3) where each f_i is a formula in the language for Algo.

With this notion in hands we can ask which type of player performs better against which other types. A fully analysis is well beyond the scope of this thesis and we leave for further research the study between player types.

5.5 A note on implementing Algo

On trying to study different strategies for Algo we propose a simple implementation in Haskell (a functional programming language).

Despite the language proposed in this chapter we simplify the Haskell program using static interpretation to calculate the probability distribution over states. The idea is to calculate the function T_a from Equation 5.1 in each moment of the game, instead calculating it at the initial state and update it as the game changes. We conjecture that both implementations - the static in this paper and a theoretical dynamic implementation as defined in this chapter - are equivalent.

We explore the implementation with some details in the appendix, and also present the whole modules in Haskell.

Chapter 6

Final Words

> Gosto dos epitáfios; eles são, entre a gente civilizada, uma expressão daquele pio e secreto egoísmo que induz o homem a arrancar à morte um farrapo ao menos da sombra que passou.
>
> ---
> Machado de Assis

> I like epitaphs; among civilized people they are an expression of that pious and secret selfishness that induces men to pull out of death a shred at least of the shade that has passed on
>
> ---
> Machado de Assis

I finish this thesis summing up the limitations and advantages of conditional probability logic and its extensions.

Regarding the proof of completeness, it is unknown if the conditional probability logic as presented (system L_{cb}) is decidable. In Zhou (2009) the system of probability logic was modified and the decidability was given. It is an open question if a similar approach is possible for conditional probability logic.

Dynamics with uncertainty of events turned out to be a difficult question and more investigation is needed. The proponents of Halpern's language have languages for uncertainty on events, although not for conditional probability spaces. A natural step is to combine both ideas. A Halpern style language complete with respect to conditional probability spaces is likely to be possible.

Many problems were solved within the present work. For instance, the extension of conditional probability logic for a quantitative degree of belief. Also, the language is complete with respect to infinite (and not necessary discrete) spaces.

The language made possible the introduction (in an Aumann style) of dynamic operators, like PAL, complete with a set of reduction axioms. The axioms were also possible because of the language and its expressive power.

On a general level, the conditional probability logic made it possible to express announcements of propositions that were believed to be false (the formula $M_0 p \wedge p \wedge [!p]L_r p$ with $r > 0$ is satisfiable). This is a contribution to the logic community itself, since neither Aumann's nor Halpern's proponents did not have a way to treat this issue.

It is the first time that Algo is formally studied. We were able to express some strategies for the game using an extension of conditional probability logic.

Appendix A

Implementing Algo in Haskell

> Isso de a gente querer ser exatamente o que a gente é, ainda vai nos levar além.
>
> Paulo Lemiski.

> That of us to want to be exactly what we are, still will take us beyond.
>
> Paulo Leminski

The following appendixes present the modules for automatic players of Algo. In this chapter we assume that the reader has some familiarity with the functional programming language Haskell. For an introductory text of Haskell from a mathematical and logical point of view see Doets and van Eijck (2004).

The module `AlgoCards` defines the types for cards, deck, players and guesses. Also it defines the functions to shuffle the cards and the instances to `show` function. We spend just a few lines explaining this module as it is straightforward.

The module `AlgoEpistemic` defines functions to (1) draw a card; (2) to compute results after guesses and (3) to calculate a guess. Also in this module we define data types for formulas of conditional probability logic and satisfaction relation for those formulas. This last part is inspired in the technical report van Eijck (2013), which defines in Haskell the language for probability belief and knowledge (S5) intending to check if formulas are satisfied or not in a given (probabilistic-)model.

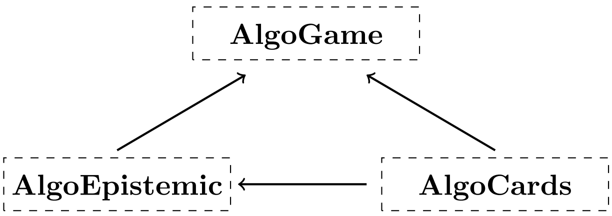

Figure A.1: Algo program module dependence. The arrow directs to modules which relies on other modules.

Module `AlgoGame` defines the function called to play the game. Also it is where we define the function for each player's turn. Most of this module is also straightforward.

Module `AlgoGame` relies on both modules `AlgoEpistemic` and `AlgoCards`. The module `AlgoEpistemic` makes use of functions from module `AlgoCards`. And finally, the module *AlgoCards* stands by itself. Figure A.1 illustrates the dependence among the modules.

A.1 AlgoCards

For the module `AlgoCards` we import the modules `Data.List` and `Monad.Random` for some operations over lists and for the function that shuffle the cards. The block of code is presented below.

```
module AlgoCards where

import Data.List
import Control.Monad.Random
```

A card has type `Card` and is a pair of an integer and a color. A color has type `Color` with constructors `Black` and `White`. Note that because of the cosntruction we have that

`(Black < White) == True`.

Next we define the types for card and hand of player. Also we define the instances to the `show` function.

```
data Card = Card { value :: Int,
                   color :: Color
                 } deriving (Eq, Ord)
```

```
data Color = Black | White  deriving (Eq,Ord)

instance Show Color where
   show Black = "B"
   show White = "W"

instance Show Card where
   show (Card a b) = show a ++ show b
```

For simplicity the type of a player is `Player` represented by an integer. A guess has type `Guess` and is a tuple (`Int, Card, Player, Player`) where the first player is the attacker and the second is the defender.

The data type `AlgoState` is the type of an Algo state. `pile` represents the pile on the game, `hand1` is the hand of player 1 and `hand2` is the hand of player 2.

A model (`Model`) is a tuple consisting of its support (`supp`), the actual state (`s0`) and two lists of boolean values (`face`) to denote which cards are turning face up or down. A model corresponds to the epistemic structure in Chapter 5.

```
type Player = Int

data Guess = Guess { pos::Int,
                     gCard::Card,
                     at::Player,
                     def::Player
                   } deriving (Show, Eq)

data AlgoState = AlgoState { pile::[Card],
                             hand1 :: [Card],
                             hand2:: [Card]
                           }

data Model = Model { supp::[AlgoState],
                     s0::AlgoState,
                     face::([Bool], [Bool])
                   }
```

The next block of code defines some basic graphic vizualization of a state of the game. We use American Standard Code (ASCII) symbols to show only the actual state of a given model.

When dysplaying, the cards on the top and bottom represents the hand of player 2 and player 1. The letters ‘U’ and ‘D’ on top of each card let us know if a card is turning

```
The next card on the pile is:
 ---
|   |\
| W ||
|___||
 \___|
 *Player2:
 _D_   _D_   _U_   _U_   _U_   _U_   _D_
|   | |   | |   | |   | |   | |   | |   |
| 0W| | 3B| | 3W| | 4B| | 5B| | 6B| | 7W|
|___| |___| |___| |___| |___| |___| |___|

 *Player1:
 _U_   _D_   _D_   _U_   _U_   _U_   _U_   _U_
|   | |   | |   | |   | |   | |   | |   | |   |
| 0B| | 1B| | 6W| | 9B| | 9W| |10B| |10W| |11W|
|___| |___| |___| |___| |___| |___| |___| |___|
```

Figure A.2: Vizualization of an Algo state during the game.

face up or down. The letters 'W' and 'B' stands for colors'white' and 'black.' Figure A.2 illustrate a situation.

```
instance Show Model where
         show m = "The next card on the pile is:"++"\n"++
                  (c (showFirstOnPile (s0 m)) ) ++"\n"++
                  " Player2 Hand:" ++"\n"++
                  (asciiHand2 (hand2$ s0 m) (face m)) ++
                  "\n"++"\n"++
                  " Player1 Hand:"++"\n"++
                  (asciiHand1 (hand1 $ s0 m) (face m))
```

```
showFirstOnPile s = if null $ pile s
                    then "X"
                    else show $ color $ head $ pile s

-- c Shows the color of the card on top of the pile.
c :: [Char] -> [Char]
c v  = " ___ " ++ "\n"
```

```
     ++"|    |\\"++"\n"
     ++"|"++ adjustSize v ++"||"++"\n"
     ++"|___||"++"\n"
    ++" \\___| "

asciiHand1 h f = (line1 (fst f))++"\n"
                 ++ (line2 h)++"\n"
                 ++ (line3 h)++"\n"++ (line4 h)

asciiHand2 h f = (line1 (snd f))++"\n"
                 ++ (line2 h)++"\n"
                 ++ (line3 h)++"\n"++ (line4 h)
```

```
line1 [] =""
line1 (f:fs) = " _"++face++"_ " ++ " " ++ line1 fs
             where face = if f then "U" else "D"

line2 [] = ""
line2 (h:hs) = "|    |" ++ " "++ line2 hs

line3 [] = ""
line3 (h:hs) = "|"++ (adjustSize $ show h) ++"|" ++
                " "++ line3 hs

adjustSize v = case length v of
                   1 -> " "++ v ++ " "
                   2 -> " "++ v
                   _ -> v

line4 [] = ""
line4 (h:hs) = "|___|"++ " " ++ line4 hs
```

The last block defines the deck of cards and the functions to shuffle the cards used to initialize the game.

```
deckN n = [Card a b | a <- [0..(n-1)], b <- [Black, White]]
deck = deckN 12

oneRandomCard :: StdGen -> [Card] -> (Card, StdGen)
oneRandomCard g d =((last $ take n d), g1 )
               where (n,g1) = randomR (1, length d) g

shuffle :: StdGen -> [Card] -> [Card]
shuffle g [] = []
shuffle g d  = c : shuffle g' d'
```

```
        where (c, g') = oneRandomCard g d
              d' = d\\[c]

shuffleDeck d = do
         seed <- newStdGen
         return $ shuffle seed d

pdeck = subsequences deck
p4deck = filter (\x -> (length x == 4)) pdeck
```

A.2 Algo Epistemic

The module `AlgoEpistemic` is divided in two parts. In the first part we define the conditional probabilistic belief language with public announcement for Algo. On the second part we define the actions of Algo.

We need the module `AlgoCards` for this module as declare below.

```
module AlgoEpistemic where

import AlgoCards
import Data.List
```

Remember that the language for conditional probability logic is given as follows:

$$\phi \doteq \top \mid p \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi \mid L^i_r(\phi||\phi) \mid [!\phi]\phi$$

where p is in the set of propositional variables.

With that in mind we define the type of formulas to be `F p`. Next we explicitly define this type and its instance of `show` function.

```
data F p = Top | P p | Ng (F p) | V [F p] | A [F p] | L Int Rational (F p)
    (F p) | PA (F p) (F p)

instance (Show a) => Show (F a) where
          show Top = "T"
          show (P a) = show a
          show (Ng a) = "-"++show a
          show (V a) =  "V"++ show (map show a)
          show (A a) = "."++ show (map show a)
          show (L i r x y) = "L"++(show i)++"_"++show r
                             ++ show x ++"//"++ show y
          show (PA a b) = "[!"++show a ++"]"++show b
        -- show (Ef e a) = "[E,"++ show e ++"]"++ show a
```

Remember that, if $\mathbb{C}$ is the set of cards in $Algo_n$ and $\mathbb{A}$ is the set of players, then the set of propositions for the language of Algo is given by

$$P = P_c \cup P_f,$$

where $P_c = \{cp_i : c \in \mathbb{C}, p \in \mathbb{A}, \text{and } i \leq n\}$ and $P_f = \{U_{p_i}, D_{p_i} : i \leq n,\ p \text{ is a player}\}$.

In our module a proposition has type `Prop` and it is either a card with an integer and a player (`PropC Card Int Player`) or a boolean, an integer and a Player (`PropF Bool Int Player`).

```
data Prop = PropC { cv::Card,
                    handPos::Int,
                    player::Player} |
            PropF { faceUp::Bool,
                    fPos::Int,
                    fPlayer::Player}

instance Show Prop where
         show (PropC c i j) = show c ++ show i ++ show j
         show (PropF b i j) = show b ++ show i ++ show j
```

We want to decide if a formula ϕ is satisfied in a given epistemic structure $\mathcal{M}$ of a game of Algo. Technically we should define the satisfaction relation on a conditional probabilistic model. However, we are going to define it on an epistemic structure. The reason being, as we defined in Chapter 5, given a epistemic structure $\mathcal{M}$ we can uniquely determine a conditional probability model $\mathcal{M}_{CP}$.

Hence, given a formula `A` and an epistemic structure `m` we denote the sentence $\mathtt{m}_{CP}, s_0 \models \mathtt{A}$, where s_0 is the current state of m, by:

`m ‘satis‘ A.`

We define the function `satis` on the next block.

```
satis m Top = True
satis m (p@(P (PropC c i j))) = if (length ((hand j)s)<=i)
                                then False else
                                (c == ((hand j) s)!!i)
                          where hand 1 = hand1
                                hand 2 = hand2
                                s = s0 m
satis m (p@(P (PropF b i j))) = if (length ((hand j)s)<=i) then False else
    (b == ((ord j) (face m))!!i)
                          where ord 1 = fst
                                ord 2 = snd
                                hand 1 = hand1
                                hand 2 = hand2
                                s = s0 m
satis m (Ng a) = not $ satis m a
```

```
satis m (V a) = or (map (satis m) a)
satis m (A a) = and (map (satis m) a)

satis m (L i r a b) = ( toRational (length $ map (\x-> satis (pointed m x
    ) (A [a, b])) univ) / toRational( length $ map (\x-> satis (pointed m x
    ) b) univ) ) >= r
                where univ = supp m
                      aNb = Ng (V [Ng a, Ng b])
satis m (PA a b) = if (satis m a) then (satis m' b) else True
              where m' = Model supp' (s0 m) (face m)
                    supp' = (filter (\x-> satis (pointed m x) b) (supp m))
```

If we want to verify the validity of a formula in a particular state other than the current state we use the function `satisAt`. If `m` is a epistemic structure, `s0` is its current state and `A` is a formula, then the following identity holds:

`(m ‘satis‘ A) == (satisAt m A s0 ).`

The set $[\![\mathtt{A}]\!]_{\mathtt{m}}$ of states in `m` that satisfies `A` is given by the function `truthSet`.

```
satisAt m phi s = satis (pointed m s) phi

pointed m s = Model (supp m) s (face m)

truthSet phi m = filter (satisAt m phi) (supp m)
```

A.2.1 Game Actions

We consider now the actions in the game.

First we define the data type `GameState`, which is a model with a list of cards representing the history of the cards in the game, i.e., if `game` has type `GameState`, then if it is not empty the function

`head (cHist game)`

has type `Card` and its value is the last card drawn (it is not defined if `cHist game` is empty).

```
data GameState = GameState { model::Model,
                             cHist::[Card]
                           }
```

Deciding the guesses Deciding which guess to announce is the first decision a player has to make. In what follows the function `calculateGuess` receives a `Player` and a `GameState` and extracts which cards from the opponent's hand is face up.

The function `calculateTGuessAI` returns a pair (`guess,p`) of type (`Guess, Int`), where the integer `p` denotes the probability of the guess `guess` being correct if Player `i` announces the guess in the given game state. Also, this part is such that for any other possible guess the value of `p` is maximum.

```
calculateGuess i game = do
                          let fs = face $ model game
                          let fi = if (i == 1) then snd fs else fst fs
                          let s = s0 $ model game
                          let ws = supp $ model game
                          let stateE = filter (\x -> r i s x fs) ws
                          let unknown = findIndices (\x -> x == False) fi
                          (guess, p) <- calculateTGuessAI i s stateE unknown
                          return (guess,p)

calculateTGuessAI ag s stateE unknowns = do
                 let handOpp = if (ag == 1) then hand2 else hand1
                 let att = ag
                 let def = head ([1,2]\\[ag])
                 let t = [((handOpp e)!!i, i)| i <- unknowns, e <-stateE]
                 let ts = counting t
                 let x = maximumBy biggEr ts
                 let guess = Guess (snd $ fst x) (fst $ fst x) att def
                 let p = snd x / (fromIntegral $ length t )
                 return (guess,p)
                      where biggEr = (\a-> \b-> (snd a) `compare` (snd b))

counting [] = []
counting (x:xs) = [(x, instanc x (x:xs))] ++ counting (delete x xs)

instanc x [] = 0
instanc x (y:ys)
  | x==y = 1+(instanc x ys)
  | otherwise = instanc x ys
```

Deciding to keep attacking Another moment in which the player has to make a choice is when he/she guesses correctly. In that case the choice is to keep attacking or pass the turn. The function `decideToKeepAttack` receives a `GameState` a `Guess` and a rational number. The idea is to allow any desirable property to be implemented in order to test different types of players.

As the code is presented player 1 keeps attacking if it holds that

$$\mathtt{m_g}, \mathtt{s_0} \vDash L^1_{0.3}(cd_i),$$

where $\mathtt{m_g}$ is the model given by `g` of type `GameState` and $\mathtt{s_0}$ is the actual state. Player 2 keeps attacking if $p \geq 0.3$.

As we saw in the function `calculateTGuessAI`, the rational `p` is calculate in a way that both functions are equivalent. In other words, the strategy of player 1 for the function `decideToKeepAttack` is redundant and should be avoided if performance of the program is in question. We wrote this strategy for player 1 to make this point clear.

```
decideToKeepAttack g (Guess i c 1 d) p =
                     (model g) `satis` L 1 0.3  (P (PropC c i d)) (Top)
decideToKeepAttack g (Guess i c 2 d) p = p>= 0.3
```

Make a Guess The last action that we modeled is the guessing action. During this event both players should act. The attacker announces the guess and the defender updates his/her view of the world given the guess.

The function `makeGuess` receives a player, a guess `g@(Guess pos c a d)`, a model `m@(Model ws s fs)` and a card `lc`. Remember that the notation given by `@` is a short way to define `ws,s,fs` such that `supp m == ws, s0 m == s, face m == fs`; and similarly for the guess `g`.

If `m, s` $\vDash cd_{pos}$ holds then the guess is successful and the function `makeGuess` returns

`(Model ws' s fs', True)`,

or else it returns

`(Model ws'' s fs'', False)`.

The model that is returned if the guess is correct is given as follows: `ws'` $= [\![cd_{pos}]\!]_{\mathtt{m}}$, and `m'` = (`ws, s, fs'`) with `fs'` equal to `fs` except to the position of `lc` in which `fs'` is `True`texttt`pos` of `c` in which `fs'` is `True`. The idea is that the card that was guessed turns face up and all the states in which the defender does not hold the card according to the guess is deleted from the model.

If the guess is incorrect the model changes as follows:

- ws1 = $[\![\neg \bigvee_{i\leq n} ca_i]\!]_{\texttt{m}}$,
- ws2 = $[\![\neg cd_{pos}]\!]_{\texttt{m'}}$ with m' == Model ws1, s, fs,
- ws'' = $[\![(lc)1_{posLast}]\!]_{\texttt{m''}}$ with m'' == Model ws2, s, fs''.

Where fs'' equal to fs except to the last card drawn's position posLast in which fs'' is True. The idea is that if the guess is unsuccessful the last card that was drawn should be turned face up, the card that was guesses is assumed not to be in the hand of the attacker nor in the position guessed.

```
makeGuess 1 g@(Guess pos c a d) m@(Model ws s fs) lc =
            if m `satis` (P (PropC c pos d)) then ((Model ws' s fs'), True)
                else ((Model ws'' s fs''), False)
            where
                    fs' = (fst fs,  (take pos (snd fs))++ [True] ++ drop (pos
                        +1) (snd fs))
                    ws' = truthSet (P $ PropC c pos d) (Model ws s fs')
                    ws1 = truthSet (Ng $ V [ (P $ PropC c i a)| i <- [0..11]])
                        m
                    ws2 = truthSet (Ng $ P $ PropC c posLast d) (Model ws1 s fs
                        )
                    fs''= ((take posLast(fst fs))++[True]++drop(posLast +1)(fst
                         fs),snd fs)
                    ws'' = truthSet (P $ PropC lc posLast 1) (Model ws2 s fs'')
                    posLast= head [i |i<-[0..11], m `satis` (P $ PropC lc i 1)]
makeGuess 2 g@(Guess pos c a d) m@(Model ws s fs) lc = if m `satis` (P (
    PropC c pos d)) then ((Model ws' s fs'), True) else ((Model ws'' s fs
    ''), False)
            where
                    fs' = ( (take pos (fst fs))++ [True] ++ drop (pos +1) (fst
                        fs),snd fs)
                    ws' = truthSet (P $ PropC  c pos d) (Model ws s fs')
                    ws1 = truthSet (Ng $ V [(P $ PropC c i a)| i <- [0..11]]) m
                    ws2 = truthSet (Ng $ P $ PropC c posLast d) (Model ws1 s fs
                        )
                    fs''= (fst fs,(take posLast (snd fs))++[True]++drop(posLast
                        +1)(snd fs))
                    ws'' = truthSet (P $ PropC lc posLast 2) (Model ws2 s fs'')
                    posLast = head [i |i<-[0..11], m `satis` (P $ PropC lc i 2)
                        ]
```

Drawing a card The drawCard function is a simplified version of the product update explained in Chapter 4. If the pile is not empty this function returns a new model where each new state is derived from a state in the previous model with the last card that was drawn added in the appropriate position.

```
drawCard 1 gs = if (null $ pile $ s0 m)
                    then gs
                    else GameState (Model (fil $ drawCardU 1 (s0 m) (supp m)
                         lc index) (drawCardS 1 (s0 m)) (drawCardF 1 (face m
                       ) index)) (lc:cH)
                         where lc = head $ pile $ s0 m
                               h1 = sort $ (hand1 $ s0 m)++[lc]
                               index = head $ findIndices (\x -> x == lc) h1
                               fil = filter (not . null . hand1)
                               m = model gs
                               cH = cHist gs
drawCard 2 gs = if (null $ pile $ s0 m)
                    then gs
                    else GameState (Model (fil $ drawCardU 2 (s0 m) (supp m)
                         lc index) (drawCardS 2 (s0 m)) (drawCardF 2 (face m
                       ) index)) (lc:cH)
                         where lc = head $ pile $ s0 m
                               h2 = sort $ (hand2 $ s0 m)++[lc]
                               index = head $ findIndices (\x -> x == lc) h2
                               fil = filter (not . null . hand2)
                               m = model gs
                               cH = cHist gs

drawCardF 1 fs index = ((take index (fst fs))++[False]++(drop index (fst fs
    )),snd fs)
drawCardF 2 fs index = (fst fs,(take index (snd fs))++[False]++(drop index
    (snd fs)))

drawCardS 1 s = AlgoState (tail p) (sort((hand1 s)++[head p])) (hand2 s)
                where p = pile s
drawCardS 2 s = AlgoState (tail p) (hand1 s) (sort((hand2 s)++[head p]))
                where p = pile s



-- (drawCardU 1 (s0 m) (supp m) lc index)
drawCardU _ _ [] _ _ = []
drawCardU 1 s (u:us) lastCard posLastCard  = (addCard 1 u posLastCard
    samecolor) ++ (drawCardU 1 s us lastCard posLastCard)
        where samecolor = sort (filter (\x -> color x == color lastCard)
            (deck \\ (hand2 u)))
drawCardU 2 s (u:us) lastCard posLastCard  = (addCard 2 u posLastCard
    samecolor) ++ (drawCardU 2 s us lastCard posLastCard)
```

```
          where samecolor = sort (filter (\x -> color x == color lastCard)
              ((deck \\ (hand1 u))))


addCard _ _ _ []      = []
addCard 1 u i (c:cs) = AlgoState (pile u) (check (hand1 u) i c) (hand2 u) :
     addCard 1 u i cs
addCard 2 u i (c:cs) = AlgoState (pile u) (hand1 u) (check (hand2 u) i c) :
     addCard 2 u i cs


check e 0 c = if (c < e!!0) then c : e else []
check e i c
       | ((i == length e) && (c > e!!(i-1)))                   = (take i e)
          ++ [c] ++ (drop i e)
       | (i < length e) && ((c > e!!(i-1)) && (c< e!!((i)))) = (take i e) ++
           [c] ++ (drop i e)
       | otherwise = []
```

For last we state a few technical functions that we used in the code.

```
r 1 s t fs = (hand1 s == hand1 t) && (consistentWith (hand2 s) (hand2 t) (
    snd fs))
r 2 s t fs = (hand2 s == hand2 t) && (consistentWith (hand1 s) (hand1 t) (
    fst fs))




--consistentWith :: [Hand] -> [Hand] -> Bool
consistentWith _ _ [] = True
consistentWith [] _ _ = True
consistentWith _ [] _ = True
consistentWith (x:xs) (y:ys) (f:fs) = if cardEqual x y f
                                  then consistentWith xs ys fs
                                  else False

--cardEqual:: Hand -> Hand -> Bool
cardEqual c1 c2 f = if f
                     then (c1 == c2)
                     else (color c1) == (color c2)
```

```
initiateUniverse d=([AlgoState s h1 h2 | h1<-p1deck, h2 <- p2deck, null (
    intersect h1 h2)], AlgoState s h1 h2)
            where h1 = sort (take 4 d)
                  h2 = sort (take 4 $ drop 4 d)
                  p1deck = [a | a <- p4deck, color (a!!0) == cc (h1!!0),
                      color (a!!1) == cc (h1!!1), color (a!!2) == cc (h1
                      !!2), color (a!!3) == cc (h1!!3)]
                  p2deck = [b | b<- p4deck, color (b!!0) == cc (h2!!0),
                      color (b!!1) == cc (h2!!1), color (b!!2) == cc (h2
                      !!2), color (b!!3) == cc (h2!!3)]
                  s = (drop 8 d)
                  cc =  color
```

Appendix B

Algo Logical

This module is the implemattation of the game of Algo. Since it is straightforward we skip the explanation and present only the code.

```
module AlgoLol where

import Data.List
import Control.Monad.Random
import AlgoCards
import AlgoEpistemic
import Control.Monad

algoGame = do
            d <- shuffleDeck deck
            let (univ, stateI) = initiateUniverse d
            let fs = ([False,False, False, False], [False, False, False,
                False])
            let m = Model univ stateI fs
            let cH = []
            let gS = GameState m cH
            putStrLn $ show m
            i <- algoLoop gS
            return i

algoLoop gs@(GameState m cH) = do
                             if (not $ endGame m)
                                then
                                 do
                                    gs' <- algoTurnI gs
                                    if (not $ endGame $ model gs')
                                       then
```

```
                                        do
                                            gs'' <- algoTurnII gs'
                                            algoLoop gs''
                                        else return 1 --Player 1 Wins
                                else return 0 -- Player 2 Wins

endGame m@(Model _ _ fs) = (all (==True) (fst fs)) || (all (==True) (snd fs
    ))

algoTurnII gs@(GameState m cH) = do
                    let game = drawCard 2 gs
                    (guess,p) <- calculateGuess 2 game
                    let (m',gResult) = makeGuess 2 guess (model game) (head $
                         cHist game)
                    if gResult && (not $ endGame $ m') then algoTurnII' (
                        GameState m' (cHist game)) else return $ GameState m'
                         (cHist game)

algoTurnII' gs@(GameState m cH) = do
                    (guess, p) <- calculateGuess 2 gs
                    if decideToKeepAttack gs guess p
                      then
                       do
                       let (m', gResult) = makeGuess 2 guess m (head cH)
                       if gResult && (not $ endGame $ m') then algoTurnII' (
                           GameState m' cH) else return (GameState m' cH)
                      else return (GameState m cH)

algoTurnI gs@(GameState m cH) = do
                    let game = drawCard 1 gs
                    putStrLn $ show $ model game
                    (guess,p) <- calculateGuess 1 game
                    let (m',gResult) = makeGuess 1 guess (model game) (head $
                         cHist game)
                    if gResult && (not $ endGame $ m') then algoTurnI' (
                        GameState m' (cHist game)) else return $ GameState m'
                         (cHist game)

algoTurnI' gs@(GameState m cH) = do
                    (guess, p) <- calculateGuess 1 gs
                    if decideToKeepAttack gs guess p
                      then
                       do
                       let (m', gResult) = makeGuess 1 guess m (head cH)
```

```
            if gResult && (not $ endGame $ m') then algoTurnI' (
                GameState m' cH) else return (GameState m' cH)
           else return (GameState m cH)
```

Bibliography

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

Alexandru Baltag and Sonja Smets. Probabilistic dynamic belief revision. *Synthese*, 165 (2):179–202, 2008.

Alexandru Baltag, Lawrence S Moss, and Slawomir Solecki. The logic of public announcements, common knowledge, and private suspicions. In *Proceedings of the 7th conference on Theoretical aspects of rationality and knowledge*, pages 43–56. Morgan Kaufmann Publishers Inc., 1998.

Pierpaolo Battigalli and Marciano Siniscalchi. Hierarchies of conditional beliefs and interactive epistemology in dynamic games. *Journal of Economic Theory*, 88(1):188–230, 1999.

Johan van Benthem, Jan van Eijck, and Barteld Kooi. Logics of communication and change. *Information and computation*, 204(11):1620–1662, 2006.

Darse Billings, Aaron Davidson, Jonathan Schaeffer, and Duane Szafron. The challenge of poker. *Artificial Intelligence*, 134(1):201–240, 2002.

Patrick Blackburn, Maarten De Rijke, and Yde Venema. *Modal Logic*, volume 53. Cambridge University Press, 2002.

Japanese Arithmetics Olympic Committee. *Atama no Yokunaru Pazuru Arugo Sansū*, volume 1. Gakken, 2011.

Kees Doets and Jan van Eijck. The haskell road to logic. *Maths and Programming, London: KingÕs College Publications*, 2004.

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

Ronald Fagin, Joseph Y Halpern, and Nimrod Megiddo. A logic for reasoning about probabilities. *Information and computation*, 87(1):78–128, 1990.

Valentin Goranko and Solomon Passy. Using the universal modality: gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992.

Paul Richard Halmos. *Measure theory*, volume 2. van Nostrand New York, 1950.

John C Harsanyi. Games with incomplete information played by 'bayesian' players, i-iii part i. the basic model. *Management science*, 14(3):159–182, 1967.

Aviad Heifetz and Philippe Mongin. Probability logic for type spaces. *Games and economic behavior*, 35(1):31–53, 2001.

George Edward Hughes and Maxwell John Cresswell. *A new introduction to modal logic*. Psychology Press, 1996.

Barteld P Kooi. Probabilistic dynamic epistemic logic. *Journal of Logic, Language and Information*, 12(4):381–408, 2003.

Barteld P Kooi. Expressivity and completeness for public update logics via reduction axioms. *Journal of Applied Non-Classical Logics*, 17(2):231–253, 2007.

Roger B Myerson. Multistage games with communication. *Econometrica: Journal of the Econometric Society*, pages 323–358, 1986.

Malempati Madhusudana Rao. *Conditional measures and applications*, volume 271. CRC Press, 2010.

Alfréd Rényi. On a new axiomatic theory of probability. *Acta Mathematica Hungarica*, 6(3):285–335, 1955.

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

Johan van Benthem. Conditional probability meets update logic. *Journal of Logic, Language and Information*, 12(4):409–421, 2003.

Johan van Benthem. *Modal logic for open minds*. Center for the Study of Language and Information, 2010.

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

Wiebe van der Hoek and Marc Pauly. 20 modal logic for games and information. *Studies in Logic and Practical Reasoning*, 3:1077–1148, 2007.

Hans van Ditmarsch. Knowledge games. *Bulletin of Economic Research*, 53(4):249–273, 2001.

Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Pieter Kooi. *Dynamic epistemic logic*, volume 337. Springer, 2007.

Hans P van Ditmarsch, Wiebe van der Hoek, and Barteld P Kooi. Dynamic epistemic logic with assignment. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 141–148. ACM, 2005.

Jan van Eijck. Demo-s5. Technical report, Technical report, CWI, Amsterdam, 2013.

Brian Weatherson. Stalnaker on sleeping beauty. *Philosophical studies*, 155(3):445–456, 2011.

Chunlai Zhou. *Complete deductive systems for probability logic with application to harsanyi type spaces*. Indiana University, 2007.

Chunlai Zhou. A complete deductive system for probability logic. *Journal of Logic and Computation*, 19(6):1427–1454, 2009.

Chunlai Zhou. Probability logic for harsanyi type spaces. *Logical methods in computer science*, 10:1–23, 2014.