The Monty Hall Dilemma

Barteld P. Kooi

Supervisors:
prof.dr. E.C.W. Krabbe
Dr. L.C. Verbrugge

Referee:
Dr. J.A. van Eck

April 1999
Department of Philosophy
University of Groningen
Contents

List of Definitions ii
Acknowledgements iii
Introduction 1

1 Probabilistic Epistemic Logic 4
1.1 Epistemic Logic ........................................ 4
1.2 Multi-Agent Systems ................................... 9
1.3 Probability Spaces ...................................... 12
1.4 Adversaries ............................................. 13
1.5 Probability for Agents ................................. 15
1.6 Evaluation ............................................. 23

2 Dynamic Epistemic Logic 26
2.1 Dynamic Logic .......................................... 26
2.2 Non-well-founded Sets ................................ 28
2.3 Possibilities and Kripke Models ....................... 30
2.4 Updates ................................................. 33
2.5 An Analysis of the Monty Hall Dilemma ............ 38
2.6 Conclusion ............................................. 48

3 Probabilistic Dynamic Epistemic Logic 49
3.1 A Comparison ........................................... 49
3.2 Probabilistic Updates on Kripke Models ............. 51
3.3 Adversaries Revisited ................................ 56
3.4 Ignorance and Learning about Probability .......... 58
3.5 An Analysis of the Monty Hall Dilemma ............ 62
  3.5.1 The standard scenario ............................ 62
  3.5.2 Laziness ........................................... 67
  3.5.3 Where the experts go wrong ................. 68
3.6 Likelihoods .............................................. 70
3.7 Conclusions and Open Questions ..................... 76

Bibliography 79
List of Definitions

1.1 Language of epistemic logic ........................................ 5
1.2 Kripke models for epistemic logic ................................ 5
1.3 Truth definition for epistemic logic .............................. 5
1.4 Kripke models for multi-agent systems ......................... 10
1.5 Language of probabilistic epistemic logic .................... 16
1.6 Kripke models for probabilistic epistemic logic ............ 17
2.1 Possibilities .............................................................. 30
2.2 Bisimulation .............................................................. 31
2.3 Truth in possibilities ................................................... 32
2.4 Language of DEL ......................................................... 34
2.5 Group updates ........................................................... 35
2.6 Introspection and factivity .......................................... 37
3.1 Language of PDEKS ..................................................... 52
3.2 Truth definition for PDEKS ........................................... 53
3.3 Probabilistic updates .................................................. 54
3.4 Consistency ............................................................... 57
3.5 Likelihoods ............................................................... 70
3.6 Static truth definition for PDEL ................................. 72
3.7 Language of PDEL ........................................................ 74
Acknowledgements

Approximately one year ago I made the plan to finish this thesis at the end of the second trimester of this academic year. Thanks to the efforts of many people, you are reading it now. Here I take the opportunity to thank them.

Without my supervisors, Erik Krabbe and Rineke Verbrugge, I could not have done this. They spent a lot of their time helping me and made me spend my time as efficiently as possible. They gave very useful detailed comments on every piece of this thesis. They always took time to see me when I wanted to discuss a specific problem with them. Rineke suggested the Monty Hall Dilemma as the subject for this thesis. For me this proved to be an excellent idea. Job van Eck refereed this thesis. I would like to thank him for the confidence he had in me.

Esther supported me through the ups and downs of writing this thesis and had to endure months of absence and absentmindedness. Long before I began writing this thesis she explained the Monty Hall Dilemma to me and convinced me for the first time that the counterintuitive solution is right.

I would like to thank Jelle Gerbrandy for finishing his dissertation just in time and for sending me a copy so that I could use his findings.

To find out how people react to the Monty Hall Dilemma, the newspaper clippings Erik Krabbe, Theo Kuipers and Hans van Ditmarsch provided were very useful. There are many people I have bothered with the Monty Hall Dilemma when I was writing this thesis. I would also like to thank them all of them for their first reactions.

The 'leveronderzoek' group provided many worthwhile comments on a very early version of this thesis. I would like to thank Jeanne Peijenborg and Gert Jan van der Reijden especially. In this context I would also like to thank David Atkinson for explaining the difference between $\sigma$-rings and $\sigma$-algebras.

Riegholt Hilbrand got up at six o'clock in the morning to discuss my thesis with me. For this and for all the support he gave and interest he showed I am grateful. Eefke Meijer and I had some very useful discussions on the subject of ignorance about probability. I also found the discussions about game theory and the Monty Hall Dilemma I had with Martin van Hees to be very useful.

I would like to thank my father for making the cover illustration and Benno Ticheler for scanning it. This thesis was made with LaTeX and would have looked very different if Haak de Vries had not helped me with this program.

As this is the conclusion of my education at the department of philosophy, I would like to take this opportunity to thank all of its staff and students who made the time I spent there as wonderful as it was. I would especially like to thank Arnold, Eefke, Jan Albert, Marlies, Riegholt and Vincent.

Last but not least I would like to thank my parents for all of their support.
Introduction

The Monty Hall Dilemma is a puzzle that often leads to furious discussions. The puzzle received worldwide attention after it was discussed in a column in *Parade Magazine* by Marilyn vos Savant, who is listed in the Guinness Book of World Records for the highest IQ. In her column ‘Ask Marilyn’ she answers questions sent in by the readers.

Suppose you’re on a game show, and you’re given the choice of three doors. Behind one door is a car, behind the others, goats. You pick a door, say number 1, and the host, who knows what’s behind the doors, opens another door, say number 3, which has a goat. He says to you, “Do you want to pick door number 2?” Is it to your advantage to switch your choice of doors?

Craig. F. Whitaker
Columbia, MD [v890]

The Monty Hall Dilemma got its name from the American game show host Monty Hall who hosted ‘Let’s Make a Deal.’ Other versions of this puzzle circulated at least as early as the sixties [Mos65, p.4, for example]. Answering this question, Vos Savant argued that it is to your advantage to switch. If you switch you lose in one third of the cases and win in two thirds of the cases.

This could be argued as follows. Suppose you have initially chosen the door with the car, then you should not switch. This happens in one third of the cases. Suppose on the other hand you pick a door that contains a goat, which happens in two thirds of the cases. The quiz master has to open a door that also contains a goat. He cannot open the door you picked. He has to open the door that contains the other goat. So, if you choose a door with a goat, the quiz master only has one option. After he has opened that door, the remaining unopened door you did not pick must contain the car. Therefore, if your initial choice was a door with a goat, switching will guarantee that you win the car. You pick such a door in two thirds of the cases. Hence in one third of the cases you lose by switching and in two thirds of the cases you win by switching.

Many people did not agree with this solution. They argued the chances of winning do not increase when you switch. Among them are some considered to be experts in the field of probability. Three Ph.D.’s wrote letters explaining that Vos Savant was wrong. The mathematician Paul Erdös also did not want to believe switching is to your advantage. The discussion made its way to the Netherlands after Rob van den Berg reported the discussion in the newspaper NRC-Handelsblad (May 18th, 1995). The response to his article was overwhelming. People called to say they could not sleep because they were thinking...
about the puzzle and demanded an explanation, many e-mails were sent and
the newspaper received over eighty letters. Some of these letters were published
(the 1st, 8th, and 15th of June). I found a letter by H. von Saher to be the
most surprising:

I will show with an analogous example that her thesis is incorrect.
Marilyn vos Savant wins a quiz (naturally), at the back of the stage
a wall is placed with 100 doors. She takes her stand in front of
door 1; in that case she has a 1% chance of standing in front of the
right door and there is a 99% chance that the prize is behind one
of all the other doors. Then the quiz master comes along; he opens
all doors from 2 to 99, so 98 doors in total. Behind none of these
doors is the prize. It must be behind door 1 or 100. According to
Marilyn vos Savant the whole 99% chance passes to door 100. It is
evident that this is absurd. Here too another situation has arisen
with only two alternatives and therefore equal chances for each of
the remaining doors. (NRG June 1st, 1995, my translation)

Although it seems that some of the crucial information eludes Von Saher, namely
that the quiz master only opens a door if he knows it does not contain a prize, he
apparently presents a good argument for Vos Savants thesis, instead of against it.
Suppose you pick door number one and all the other doors except door
number 53 are opened; to me it seems even more obvious that in that case you
should switch than in the three door case. These arguments seem to be difficult
to understand and can even make people quite angry. Marilyn vos Savant was
ridiculed in many of the letters written to her. There is even an entire website
dedicated to showing she is wrong in this and other matters\(^1\).

There were also letters by people who reported that they had actually played
the game or had made a computer simulation of the game and that the results
concurred with Vos Savants claim\(^2\). Theo Kuipers wrote that playing the game
with his wife not only convinced his wife, but also made them find an elegant
solution [Won91]. Yet some reported that their findings concurred with Vos
Savant, but they were still not convinced. Some even offered their computer
programs to be distributed hoping that someone could find an error. Experimental
data can give a correct answer to the question whether you should switch,
but experimental data alone do not yield a correct analysis of the problem. Yet
playing the game seems to be the final option for those who are not convinced.

The Monty Hall Dilemma is a puzzle for which intuitions fail many people.
It is surprising that these wrong intuitions are very strong. But there are many
puzzles and paradoxes where one can have strong intuitions that are wrong.
One might think for example that there are more natural numbers than there
are prime numbers, but this is not true. The best way to show that such
counterintuitive results are in fact correct is to use some formal method such as
logical analysis.

There are three different aspects of the puzzle that such a logical analysis
should be able to handle: probability, knowledge and information change.
Probability plays an important role. Vos Savants answer to the question was

\(^1\)http://www.wiskit.com/maai/maai.html

\(^2\)Several good simulations of the Monty Hall Dilemma can be found on the Internet; e.g.
in terms of probability. Another crucial element of the Monty Hall Dilemma is that the quiz master knows where the car is, therefore knowledge plays an important role. Finally the information the quiz master and the player have changes as the game progresses. A door is chosen, another door is opened. All these aspects have to be dealt with in order to give a satisfactory analysis of the Monty Hall Dilemma.

There are logics that can handle both probability and knowledge. One of these is presented in chapter 1. An analysis of the Monty Hall Dilemma is given using this logic. Yet it turns out that this analysis is not completely satisfactory, because the arguments involved in the Monty Hall Dilemma cannot be expressed in its formal language. There is also a system that can handle both knowledge and information change. An analysis using this system is given in chapter 2, but because probability is not incorporated in this logic, the analysis is not satisfactory either. Therefore in chapter 3 a combination of these two systems is given, which is used to give a third and final analysis of the Monty Hall Dilemma. It turns out that all three aspects of the puzzle can be translated into its formal language.

There are other formal methods one might use to analyse the Monty Hall Dilemma. One might use a Bayesian approach. One can also think of decision theory or game theory. I do not at all discard these methods as successful means of analysing the Monty Hall Dilemma. Yet they will play no part in this thesis. Bayesian analyses of the Monty Hall Dilemma are given in [MCDD91] and in [Wou91]. Decision theory and game theory are both concerned with expected gain, utility, strategies, and those sorts of things. The logic presented in chapter 3 is not concerned with these concepts. It models how probability that is assigned to certain events should be adapted when information is acquired.
Chapter 1

Probabilistic Epistemic Logic

In this chapter I give an analysis of the Monty Hall Dilemma using the probabilistic epistemic logic as it is presented in the article Knowledge, probability, and adversaries by J.Y. Halpern and M.R. Tuttle [HT93] and the article Reasoning about knowledge and probability by R. Fagin and J.Y. Halpern [FH94]. I will focus mainly on the first of these articles. With this probabilistic epistemic logic sentences like 'according to her it will rain with a probability of at least \( \alpha \)' can be formalised. This logic is based on another logic which is concerned with so-called 'multi-agent systems.' These systems can be seen as interacting people, but also communicating computers or a mixture of people and computers. In multi-agent systems knowledge plays an important role. One of the logics that multi-agent systems logic is in its turn based upon is epistemic logic, which is concerned with knowledge. Epistemic logic is introduced in section 1.1, which is based on [BDKM91, FHMV95, Kra89, MH95, Ver99]. The logic for multi-agent systems is presented in section 1.2. This logic is extensively discussed in [FHMV95]. In this section the first part of the analysis of the Monty Hall Dilemma, which is spread out over several sections, is presented. In section 1.3 the basic structure Halpern and Tuttle use for probability is introduced. Section 1.4 shows how to add probability to multi-agent systems and in particular how it can be added to the system of the Monty Hall Dilemma. In section 1.5 the probability agents should assign to the events in a multi-agent system is discussed. Of course the probabilities the player should assign to the events in the Monty Hall Dilemma will receive special attention. Finally, in section 1.6 I assess to what extent the analysis presented in this chapter is satisfactory.

1.1 Epistemic Logic

In Knowledge and Belief [Hin62] Jaakko Hintikka described knowledge in terms of possible worlds for the first time. If someone cannot rule out that something is the case, there is a possible world accessible to that person where it is the case. In this context possible worlds are also called epistemic alternatives. So epistemic logic is a modal logic. The difference is that in alethic modal logic the symbol ‘\( \Box \)' means 'it is necessary that,' but in epistemic logic it means 'he
1.1 Epistemic Logic

or she knows that. The symbol ‘\( K \)’ (‘Knows’) is often used instead of ‘\( \Box \)’.

Epistemic logic can be used to model situations involving more than one person. (In this context persons are also called ‘agents’.) Suppose for example that there is a situation involving two persons, \( a \) and \( b \). If the proposition ‘\( p \)’ means ‘it is raining’, then ‘\( a \) knows it is raining’ can be formalised as ‘\( K_a p \)’. The subscript indicates that we are concerned with \( a \)’s knowledge. The sentence ‘\( b \) knows that \( a \) knows it is raining’ can be formalised as ‘\( K_b K_a p \)’.

**Definition 1.1 (Language of epistemic logic)**

Let a set of propositional variables \( \mathcal{P} \) and a finite set of agents \( \mathcal{A} \) be given. The language of epistemic logic \( \mathcal{L}_{\mathcal{P}, \mathcal{A}} \) is the set of sentences such that:

1. \( p \in \mathcal{L}_{\mathcal{P}, \mathcal{A}} \) for every \( p \in \mathcal{P} \)
2. if \( \phi, \psi \in \mathcal{L}_{\mathcal{P}, \mathcal{A}} \), then \( \neg \phi, (\phi \land \psi) \in \mathcal{L}_{\mathcal{P}, \mathcal{A}} \)
3. if \( \phi \in \mathcal{L}_{\mathcal{P}, \mathcal{A}} \) and \( a \in \mathcal{A} \), then \( K_a \phi \in \mathcal{L}_{\mathcal{P}, \mathcal{A}} \)
4. only those sentences that can be generated by clauses 1–3 in a finite number of steps are elements of \( \mathcal{L}_{\mathcal{P}, \mathcal{A}} \).

Moreover \( (\phi \lor \psi) \) is an abbreviation for \( \neg(\neg \phi \land \neg \psi) \), \( (\phi \rightarrow \psi) \) is an abbreviation for \( \neg(\phi \land \neg \psi) \) and \( (\phi \leftrightarrow \psi) \) is an abbreviation for \(( (\phi \rightarrow \psi) \land (\psi \rightarrow \phi) ) \). I will also use the convention to omit the outermost parentheses of a sentence.

It was already mentioned above that epistemic logic is a modal logic. The standard semantics for epistemic logic does not differ much from standard semantics for modal logic. In standard semantics a model \( M \) is a tuple \((W, R, V)\), where \( W \) is a nonempty set of possible worlds, \( R \) is a binary relation defined on \( W \), also called an accessibility relation, and \( V \) assigns to every propositional variable \( p \in \mathcal{P} \) a subset of \( W \). Given a model the truth value of any sentence in any world can be calculated. The Kripke models for epistemic logic are very similar. The only difference is that there is more than one accessibility relation, viz. one for every agent. These models are so-called multi-modal models.

**Definition 1.2 (Kripke models for epistemic logic)**

Let a set of propositional variables \( \mathcal{P} \) and a set \( \mathcal{A} \) of agents be given. A Kripke model for epistemic logic \( M \) is a triple \((W, R, V)\) such that:

- \( W \) is a nonempty set, the set of possible worlds
- \( R \) is a set of accessibility relations that contains an accessibility relation \( R_a \subseteq W \times W \) for each agent \( a \in \mathcal{A} \)
- \( V \) is a function that assigns a set of possible worlds to each proposition \( p \in \mathcal{P} \).

A pair \((W, R)\), where \( W \) is a set of possible worlds and \( R \) is a set of accessibility relations is called a frame. The truth definition is usually:

**Definition 1.3 (Truth definition for epistemic logic)**

Let a set of propositional variables \( \mathcal{P} \), a set \( \mathcal{A} \) of agents and a Kripke model \((W, R, V)\) be given such that \( p \in \mathcal{P} \), \( w \in W \) and \( \phi, \psi \in \mathcal{L}_{\mathcal{P}, \mathcal{A}} \).

\[
\begin{align*}
w \models p & \iff w \in V(p) \\
w \models \neg \phi & \iff w \not\models \phi \\
w \models \phi \land \psi & \iff w \models \phi \text{ and } w \models \psi \\
w \models K_a \phi & \iff \text{ for every } w' \text{ such that } wR_aw' \models \phi 
\end{align*}
\]
Consider the following example. Suppose two children, $a$ and $b$, have been playing outside. Both of them can see if the other child’s face is muddy, yet they cannot see their own faces. This situation can be analysed using a Kripke model. A picture of this Kripke model is shown in figure 1.1. The possible worlds are indicated by pairs $(x, y)$, where $x$ and $y$ stand for the state $a$’s respectively $b$’s face is in. The number 0 means the child’s face is not muddy, 1 means it is muddy. The labelled arrows indicate the accessibility relations of $a$ and $b$. In $(0,1)$ for example $a$ cannot rule out her face is muddy, but she also cannot rule out she is not muddy (which is actually the case). Therefore $(0,1)$ and $(1,1)$ are both accessible to $a$. She does know that $b$’s face is muddy. Therefore in both worlds that are accessible to her $b$’s face is muddy.

Usually an axiomatization is used for epistemic logic. The simplest axiomatization is the system $K_{(m)}$, which corresponds to the system $K$ that is used for standard mono-modal logics. The system consists of the following axioms and rules.

**A1** $\vdash \phi$, if $\phi$ is valid in classical propositional logic

**A2** $\vdash K_{a}(\phi \rightarrow \psi) \rightarrow (K_{a}\phi \rightarrow K_{a}\psi)$

**MP** $\phi, \phi \rightarrow \psi \vdash \psi$

**Nec** If $\vdash \phi$, then $\vdash K_{a}\phi$

In this system every agent knows all of the logical consequences of her knowledge. Obviously this is not realistic (at least not for human knowledge). This problem is known as ‘the problem of logical omniscience’ and is one of the weak points of epistemic logic. (The problem is extensively discussed in [FHMV95].) The problem becomes apparent when we contemplate a game such as chess.
1.1 Epistemic Logic

Assuming someone knows all the rules of the game, then given the initial positions of the pieces on the chess-board, she knows all the ways in which the game could progress. This is beyond the mental capacities of ordinary (and even extraordinary) human beings.

The system is usually extended with one or more of the following axioms.

\[ \text{A3} \vdash K_a \phi \rightarrow \phi \]

\[ \text{A4} \vdash K_a \phi \rightarrow K_a K_a \phi \]

\[ \text{A5} \vdash \neg K_a \phi \rightarrow K_a \neg K_a \phi \]

The system \( K(m) \), extended with A3 is called \( KT(m) \). The system \( KT(m) \) extended with A4 is called \( S4(m) \). The system \( S4(m) \) extended with A5 is called \( S5(m) \). The system \( K(m) \), extended with A4 and A5 is called \( K45(m) \). Often I omit the subscript from these systems for brevity (\( S5 \) instead of \( S5(m) \) for instance).

Axiom A3 links up with the idea of knowledge as ‘justified true belief.’ If you know something to be the case, then it is the case. (This is also called factivity.) There is an interesting connection between this axiom and frames. If axiom A3 holds in a frame given any valuation function, then the accessibility relations of the agents are reflexive (for every \( a \in A \) and every \( w \in W \): \( wR_ag \)).

The axioms A4 and A5 have been severely criticized. A4 expresses that if someone knows something, then she knows that she knows it. (This is also called positive introspection.) If this would hold for human knowledge, everyone would be aware of everything she knows. This is not realistic. Suppose for example that a student is studying geography. She has to memorize all the capital cities of all European countries. To find out whether she knows all of them she has to test this behaviorally by writing them down on a piece of paper. But if she was aware of all of her knowledge, she would not have to do this, she would simply know it. There is also a connection between A4 and frames. If axiom A4 holds in a frame given any valuation function, then the accessibility relations of the agents are transitive (for every \( a \in A \) and every \( v, w, u \in W \): if \( vR_ag \) and \( wR_av \) then \( uR_u \)).

Axiom A5 expresses that if someone does not know something, she knows she does not know it. (This is also called negative introspection.) The example of the geography student applies to this axiom. Because for the student to find out what she does not know yet, she has to do the same test. Another example is that almost no one knows that the element protactinium (atomic number 91) with an atomic weight of 231, 03589u has a half life of \( 3.43 \times 10^4 \) years. Most people will not be aware of their ignorance regarding protactinium, because most people simply have never heard of protactinium (except maybe when they know Tom Lehrer’s song ‘The Elements’ by heart.) If axiom A5 holds in a frame given any valuation function, then the accessibility relations of the agents are euclidian (for every \( a \in A \) and every \( v, w, u \in W \): if \( vR_ag \) and \( vR_au \) then \( uR_u \)). If all the axioms of \( S5(m) \) hold in a frame, the accessibility relations are reflexive, transitive and euclidian. Some elementary set theoretical reasoning shows that such accessibility relations are equivalence relations. One can prove that the axiomatization \( S5(m) \) is sound and complete with respect to the class of models where the accessibility relations are equivalence relations. The models Halpern and Tuttle use in their article [HT93] are also \( S5(m) \) models. \( S5(m) \) is often used as a model for knowledge. The system \( K45(m) \) is often used as a
model for belief. In this view belief is knowledge without factivity. The system presented in chapter 2 mainly focuses on belief.

There are two more concepts in epistemic logic that will play a marginal role in this thesis, general knowledge and common knowledge. To incorporate these two notions in the formal language two modal operators have to be added: $E$, which stands for ‘Everybody knows’ and $C$, which stands for ‘Common knowledge.’ If everyone in a group $B$ knows that $\phi$ is the case, this can be formalised as $E_B \phi$. This sentence is equivalent to a conjunction of all the sentences that express that a member of $B$ knows that $\phi$. (Remember that the set of agents is finite.) Semantically speaking a sentence $E_B \phi$ is true in a possible world if $\phi$ is true in every world that is accessible for at least one of the members of $B$.

The concept of common knowledge stems from David Lewis’s attempt to analyse the concept of convention [Lew69]. In traffic for example it is a convention that everyone drives on the right side of the street. If this is a convention everyone knows that you are supposed to drive on the right side of the street. But it must also be the case that one can trust other people to follow this convention. Therefore everyone knows that everyone knows that you are supposed to drive on the right side of the street. The sentence $C_B \phi$ can be thought of as an infinite conjunction $E_B \phi \land E_B E_B \phi \land E_B E_B E_B \phi \land \ldots$. Semantically this means that $C_B \phi$ is true iff $\phi$ is true in every world that is accessible by the transitive closure of the accessibility relations of the members of $B$. ($\phi$ is true in every world accessible by a sequence of ‘arrows’ of the accessibility relations of the members of $B$.)

The axiomatizations of epistemic logic can also be extended such that general and common knowledge are treated. The axiomatization $KEC_{(m)}$ is an extension of $K_{(m)}$ and has two additional axioms and one additional rule.

$$C1 \vdash E_B \phi \leftrightarrow \bigwedge_{a \in B} K_a \phi$$

$$C2 \vdash C_B \phi \rightarrow E_B (\phi \land C_B \phi)$$

$$RC1 \text{ If } \vdash \phi \rightarrow E_B (\psi \land \phi), \text{ then } \vdash \phi \rightarrow C_B \psi$$

This system can also be extended to $KTEC_{(m)}$, $SAEC_{(m)}$, $S5EC_{(m)}$ and $K45EC_{(m)}$.

Axiom $C1$ expresses that $\phi$ is general knowledge for a group iff every member of that group knows that $\phi$ is the case. Axiom $C2$ expresses that $\phi$ is common knowledge for a group only if every member of that group knows that $\phi$ is the case and that $\phi$ is common knowledge. The rule $RC1$ links up to the idea that $C_B \phi$ is an infinite conjunction. Suppose that $\phi \rightarrow E_B (\psi \land \phi)$ is true. Then the sentence $\phi \rightarrow E_B (\psi \land E_B (\psi \land \phi))$ must also be true, and so forth. Every sentence of the form $E_B E_B \ldots \phi$ is implied by $\phi$. Consequently $\phi \rightarrow C_B \psi$.

Although common knowledge is a nice concept which seems to give a good idea of what a convention is, in many ordinary situations it is difficult to attain common knowledge. A well-known example of this is the so-called Byzantine General Problem. There are two generals encamped on two sides of a hill: they are allies. The enemy occupies the plain at the bottom of the hill. Neither one of the generals can defeat this enemy on his own. One of them devises the plan that they will attack simultaneously the following day at high noon, because together they will surely be victorious. He can communicate with the other general by sending a messenger across the hill. He cannot be sure this
messenger will arrive, because there is a chance he will be captured by the enemy. Therefore he asks the other general to send a messenger back, if the messenger arrives, thereby acknowledging that the message is received. However, he will also want to know whether his message is received. So a third messenger will be sent, and so on. If this is the only way the generals can communicate, they will never be sure that the other one will attack. This is because it can never be common knowledge that all the messengers have arrived.

The problem of logical omniscience, the objections raised against A4 and A5 and the problems concerning common knowledge have to be taken seriously. Epistemic logic gives an ideal picture of knowledge and cannot be applied to every situation. However it proves to be a good approach for many everyday situations, including the Monty Hall Dilemma.

1.2 Multi-Agent Systems

The term multi-agent systems is slightly vague, because it is not exactly clear which things count as agents. It seems best to take the broadest interpretation of the term and consider computer processing units, robots as well as persons as agents. The authors of *Reasoning About Knowledge* [FHMV93] seem to focus on communicating computer processors as agents of multi-agent systems. Such systems of communicating computer processors are usually called distributed systems. Consequently much of their terminology stems from computer science. The main aim of the authors is to be able to reason about what goes on in these systems from the outside. Epistemic logic as it was introduced in the previous section cannot provide a good model for multi-agent systems, because the states the agents of the system are in, must be capable of change, as a result of interaction for example.

In multi-agent systems agents can find themselves in any of a number of states. These are also called local states, because they involve only one agent. In the case of distributed systems, local states can be thought of as states of a processor's memory. But if you want to model a game like poker, for instance, the local states can be thought of as the cards an agent holds. To give a complete picture of a system the environment has to be taken into account as well. A state of the environment in a distributed system might yield information about whether a certain communication line is working or not. In case of a poker game a state of the environment might consist of the cards that are still in the deck on the table. In general there is one set of states for the environment and there is one set of local states for each agent.

\[ L_e = \{ s \mid s \text{ is a state of the environment} \} \]

\[ L_a = \{ s \mid s \text{ is a local state of agent } a \} \]

A global state of the system is nothing more than a state of the environment combined with the local states of the agents. If a system contains \( n \) agents, a global state is an \( n + 1 \)-tuple \((s_e, s_1, \ldots, s_n)\). The set of global states \( \mathcal{G} \) of a system is defined as:

\[ \mathcal{G} = L_e \times L_1 \times \cdots \times L_n \]

As was noted above, multi-agent systems are subject to change. The state of the environment changes and the local states of the agents can change. So the
global state of the system can change. One of the assumptions that is made is that time is discrete, i.e. not continuous. Although this gives a distorted picture of time as it is in reality, it is quite suitable for computer processors, because they change in discrete steps. An analysis of the Monty Hall Dilemma will not be affected by this assumption either. The natural numbers are taken to model time. There are of course many more discrete structures. The advantage of the natural numbers is that they have a clear starting point, as do many computer programs.

A real system cannot simply go from one global state to any other. The ways in which a system can develop is usually limited. Such a possible development is called a run. A run is a function from the natural number to the set of global states of the system.

\[ r : \mathbb{N} \rightarrow \mathcal{G} \]

A system \( R \) is defined to be a set of runs. The global state of a system in a run \( r \) at time \( m \) is \( r(m) \). A pair \( (r, m) \) consisting of a run \( r \) and a time \( m \) is called a point. The global state at a point \( (r, m) \) is \( r(m) \). Points will be used as the possible worlds in a Kripke model. The local state of an agent \( a \) at a point \( (r, m) \) is indicated by \( r_a(m) \) (note the subscript).

In an interpreted system the truth value of every propositional variable in a global state is defined. The function \( \pi \) assigns a valuation function to every global state. A valuation function assigns a truth value to every propositional variable in \( \mathcal{P} \).

\[ \pi : \mathcal{G} \rightarrow (\mathcal{P} \rightarrow \{0, 1\}) \]

An interpreted system \( I \) is defined to be a pair \( (R, \pi) \).

In order to introduce the concept of an agent’s knowledge in an interpreted system a Kripke model has to be defined which corresponds to the interpreted system. The set of points is taken as the set of possible worlds. The set of global states is not taken as the set of possible worlds, because a system can have the same global state in different runs and even at different times. The accessibility relation \( R_a \) of an agent \( a \) is defined as follows. A point \( (r', m') \) is accessible to \( a \) from a point \( (r, m) \) iff \( a \) has the same local state in both \( (r, m) \) and \( (r', m') \). This construction ensures that the accessibility relations are equivalence relations. Hence the models are \( \mathcal{S} \mathcal{S} \) models.

**Definition 1.4 (Kripke models for multi-agent systems)**

Let a set of propositional variables \( \mathcal{P} \) and a set \( \mathcal{A} \) of agents be given. Given an interpreted system \( I \) the Kripke model \( M_I \) is a triple \( (W, R, V) \) such that

- \( W \) is the set of points of \( I \)
- \( R \) is a set of accessibility relations that contains an accessibility relation \( R_a \subseteq W \times W \) for each agent \( a \in \mathcal{A} \) such that \( (r, m) \models R_a(r', m') \) iff \( r_a(m) = r_a(m') \)
- \( V \) is a function that assigns a set of points to each proposition \( p \in \mathcal{P} \) such that \( V(p) = \{(r, m) \mid \pi(r(m))(p) = 1\} \).

Definition 1.3 can still be used as a truth definition. The construction of this logic is represented schematically in figure 1.2.

The Monty Hall Dilemma can be viewed as a multi-agent system with two agents, the player \( p \), and the quiz master \( q \). Remember that the accessibility
1.2 Multi-Agent Systems

![Diagram](image)

**Figure 1.2:** The construction of the logic for multi agent systems

Relation of an agent was defined in such a way that a point is accessible for an agent from another point if the agent has the same local state at both points. Because the quiz master has complete knowledge of the system, all information must be encoded into his local state. His local state can be represented as a 4-tuple \((w, x, y, z)\) such that \(w, x, y, z \in \{0, 1, 2, 3\}\). Here \(w\) represents time. At time 0 nothing has happened yet, at time 1 the car is placed behind one of the doors and goats behind the others, at time 2 the player makes a choice and finally at time 3 the quiz master opens a door. In the local state of the quiz master \(x\) stands for the door with the car behind it (when \(x = 0\) the car is not behind any of the doors), \(y\) stands for the choice \(p\) makes and \(z\) stands for the door that is opened by \(q\). The player only knows the time it is, the choice he has made, and the door that is opened, therefore his local state can be represented by a triple \((t, u, v)\), where \(t, u, v \in \{0, 1, 2, 3\}\) \((t\) stands for the time, \(u\) for the choice, and \(v\) for the door that is opened.) This representation already limits the system, because there cannot be more than one car behind the doors, the player can only make one choice and only one door can be opened. A global state of this system is the player’s local state followed by the quiz master’s local state, so it is a pair \(((t, u, v), (w, x, y, z))\). (Because the quiz master has complete knowledge of the system, there is no need for a state of the environment to give a complete picture of the system.) Of course not all syntactically correct global states can occur in the system. A global state must at least satisfy the condition that \(t = w, u = y\) and \(v = z\), because the player and the quiz master must agree on the time, the choice that is made and the door that is opened.

The system can only develop in a limited number of ways. The system \(\mathcal{R}\) of runs can be represented as a so-called computation tree, which is shown in figure 1.3. To make the tree as simple as possible the global state is represented by the local state of the quiz master, because this contains all the relevant information. This representation is a bit deceptive, because the runs are merged into one tree. Remember that the same node can represent different points in different runs. The tree consists of twelve runs. Some implicit assumptions are made explicit in constructing this computation tree. For example it is assumed that the quiz master does not open a door with a car
behind it. This system can easily be made into an interpreted system. I take \( P = \{A_1, A_2, A_3, C_1, C_2, C_3, O_1, O_2, O_3\} \) as the set of propositional variables. \( A_i \) is the proposition that expresses that the car is behind door number \( i \), \( C_i \) expresses that \( p \) chooses door number \( i \) and \( O_i \) expresses that \( q \) opens door number \( i \). The valuation function \( \pi \) can be defined as follows.

\[
\begin{align*}
\pi((t, u, v)(w, x, y, z))(A_i) &= 1 \quad \text{iff} \quad x = i \\
\pi((t, u, v)(w, x, y, z))(C_i) &= 1 \quad \text{iff} \quad u = i \text{ and } y = i \\
\pi((t, u, v)(w, x, y, z))(O_i) &= 1 \quad \text{iff} \quad v = i \text{ and } z = i
\end{align*}
\]

A Kripke-model can be defined on the basis of this interpreted system. The possible worlds are the points of the runs. In general the accessibility relation \( R_a \) of an agent \( a \) is defined as \( \{(r, m), (r', m')\} | r_a(m) = r'_a(m')\). Because the quiz master has complete knowledge of the system, a point \( c \) is accessible for the quiz master if and only if those points are the same in all respects, except the run. The only difference between the quiz master’s local state and the player’s local state is that the door with the car is not encoded in the player’s local state. This means that a point \( c' \) that is accessible to \( p \) from a point \( c \), must be the same as \( c \) with respect to the time, the choice \( p \) makes and the door that is opened, but it may differ with respect to the door the car is behind. The function \( V \) can be constructed in the usual way from \( \pi \).

### 1.3 Probability Spaces

Probability spaces are set-theoretical objects that are used extensively in both the article by Fagin and Halpern [FH94] and the article by Halpern and Tuttle
1.4 Adversaries

Therefore it is important to give a precise definition. This section is based on Measure Theory by P.R. Halmos [Hal50] and [Deh95]. A probability space is a tuple \((S, \mathcal{H}, \mu)\), such that \(S\) is a set, \(\mathcal{H}\) is a \(\sigma\)-algebra on \(S\), and \(\mu\) is a probability measure. So we need three ingredients. Luckily the first one, \(S\), is simply a nonempty set, which is also called a sample space.

The next ingredient is a \(\sigma\)-algebra. A class (a set of sets) \(\mathcal{H}\) is a \(\sigma\)-algebra on \(S\) if and only if \(\cup \mathcal{H} = S\). \(S\) is an element of \(\mathcal{H}\) and \(\mathcal{H}\) is closed under the formation of differences and countable unions, formally:

- \(S \in \mathcal{H}\)
- if \(A, B \in \mathcal{H}\) then \(A - B \in \mathcal{H}\)
- if \(A_i \in \mathcal{H}\), \(i = 1, 2, 3, \ldots\), then \(\bigcup_{i=1}^{\infty} A_i \in \mathcal{H}\).

Note that \(\emptyset\) is an element of every \(\sigma\)-algebra, since \(S - S = \emptyset\), and that a \(\sigma\)-algebra is also closed under the formation of complements, since \(S - A = A'\).

The last ingredient is a probability measure. A measure \(\mu\) is a positive real-valued function defined on sets such that \(\mu(\emptyset) = 0\). Moreover it must be countably additive, that is, given a countably infinite number of pairwise disjoint sets \(A_i\) (for all \(i \neq j\): \(A_i \cap A_j = \emptyset\)) the following holds:

\[
\mu\left(\bigcup_{i=1}^{\infty} A_i\right) = \sum_{i=1}^{\infty} \mu(A_i)
\]

A probability measure is a measure defined on the \(\sigma\)-algebra \(\mathcal{H}\) of a probability space such that \(\mu(S) = 1\). Consequently the measure of an element of \(\mathcal{H}\) is in the interval \([0, 1]\).

The classic example of an experiment involving probability is a throw of a six-sided die. The sample space is the set of all possible outcomes: \(S = \{1, 2, 3, 4, 5, 6\}\). It is natural to take the powerset of \(S\) as the class of measurable subsets. This is a \(\sigma\)-algebra. Assuming the die is fair, the measure of each singleton \(\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\) and \(\{6\}\) is \(\frac{1}{6}\). Given this probability space the probability that the outcome is even for example is \(\frac{1}{2}\), because \(\mu(\{2, 4, 6\}) = \frac{1}{2}\).

This simple example shows how you can use a probability space to model an experiment.

1.4 Adversaries

It is not a simple matter to incorporate probability in multi-agent systems. One of the problems is illustrated by the following example discussed by Halpern and Tuttle.

Now consider the system \([\ldots]\) consisting of two agents, \(p_1\) and \(p_2\), where \(p_1\) has an input bit and two coins, one fair coin landing heads with probability \(1/2\) and one biased coin landing heads with probability \(2/3\). If the input bit is 0, \(p_1\) tosses the fair coin once and halts. If the input bit is 1, \(p_1\) tosses the biased coin once and halts. [HT93]
It is easy to view this system as a computation tree (see figure 1.4). \((h \text{ and } t \text{ are abbreviations for heads and tails respectively.})\) The problem is that you cannot assume a probability distribution on the input bits. Thus you cannot say what the unconditional probability of heads is. Halpern and Tuttle conclude, after discussing this problem, that some choices are inherently nonprobabilistic. In order to handle this problem they introduce the concept of an adversary. An adversary makes all nonprobabilistic choices in a system. It is best to think of this in terms of computer programs. A computer program designer is not sure who is going to use her programs. If she writes a program that does not always give the correct output, she might want to know the probability that her program does not give the correct output. This depends on the choices the user makes, who can be thought of as an adversary.

In the Monty Hall Dilemma however all choices are nonprobabilistic. Neither the player nor the quiz master make their decisions by tossing a coin or something like that. As Halpern and Tuttle use the term, an adversary would make both the player’s decisions as well as the quiz master’s decisions, though this may seem somewhat counterintuitive.

Halpern and Tuttle do not elaborate on the question what an appropriate adversary is, given a description of a system. They remark it is a choice for the system designer. The choice is not limited to one adversary. Different adversaries may be realistic. In their system an agent can consider different adversaries possible at the same time.

Once an adversary \(A\) is chosen the system can be viewed as a labelled computation tree \(T_A\). In the example given above the assumption could be made that two adversaries are appropriate; an adversary \(A\) who assigns probability \(\frac{1}{2}\) to input bit 1 and an adversary \(B\) who assigns probability \(\frac{1}{4}\) to input bit 1. The resulting labelled computation trees are shown in figure 1.5. It is assumed that the adversary is encoded in the global state of the system in order to ensure the same global state does not occur in two differently labelled computation trees. Halpern and Tuttle view this computation tree as a probability space \((\mathcal{R}_A, \mathcal{H}_A, \mu_A)\) where \(\mathcal{R}_A\) is the set of runs. \(\mathcal{H}_A\) is constructed by taking sets of runs with a common finite prefix and closing under countable union and complementation (consequently \(\mathcal{H}_A\) is a \(\sigma\)-algebra on \(\mathcal{R}_A\)). (Two runs \(r\) and \(r'\) have the same prefix up to point \(k\) if \(r(i) = r'(i)\) for every \(i \leq k\).) \(\mu_A\) is a probability measure on \(\mathcal{H}_A\). In case of the Monty Hall Dilemma all runs are finite which means \(\mathcal{H}_A\) is the powerset of \(\mathcal{R}_A\), therefore every set of runs is measurable. The probability assigned to a set of runs with a common prefix by \(\mu_A\) is defined as
the product of the probabilities labelling the edges of the common prefix. The probability of a finite run is the product of the probabilities labelling the run.

To add probability to the multi-agent system that was constructed for the Monty Hall Dilemma an appropriate set of adversaries has to be chosen. As was noted before the adversary makes both the player's choices as well as the quiz master's choices. I assume that the most realistic adversary does not have any preferences regarding the choices the agents make. Hence the probability it assigns to the transitions in the tree are such that if the player or the quiz master have to make a choice the probability assigned to the alternatives is the same. This labelled computation tree is shown in figure 1.6. To be precise the adversary should also be encoded into the global state into the system. But because only one adversary is being considered it has been omitted.

Are there any other adversaries that should be considered? Can we simply assume that the probability the car is behind one door is equal to the probability that the car is behind another door? Maybe the stagehand is a bit lazy and will drive the car to the door that is nearest to her. Perhaps the rutting season has just begun and it seems wise to the stagehand to keep the goats as far apart as possible. Can we simply assume that the player does not have any preferences regarding the doors? Perhaps he has a lucky number. Perhaps he is also lazy and will choose the door closest to him. Can we simply assume that the quiz master does not have any preferences when opening the doors? Maybe he is also lazy and will walk to the door closest to him. Maybe he likes to walk and will always go to the door that is the farthest away from him. In section 3.5.2 I will return to these questions. In another article [Hal95] Halpern indicates which computation trees can be considered according to him. This is briefly discussed in section 3.7. For now I will let these considerations be and only consider the labelled computation tree of figure 1.6.

1.5 Probability for Agents

The system developed so far does not yet enable us to say anything about the probability an individual agent ascribes to a certain event. Because this may vary over time, a probability space must be assigned to each agent at each point. Let $\mathcal{P}$ be a mapping from agents $a$ and points $c$ to probability spaces $\mathcal{P}_{a,c} = (\mathcal{S}_{a,c}, \mathcal{H}_{a,c}, \mu_{a,c})$. Our task is to say which probability space assignments are appropriate, given a system and a set of adversaries. With a probability space
assignment a truth definition can be given. This will be discussed first. Then the construction of probability space assignments from sample space assignments will be discussed. Finally different sample space assignments will be discussed.

Once an appropriate probability space assignment is constructed, we can make sense of sentences like “according to agent $a$ $\phi$ is true with probability at least $\alpha$.” This can be formalized as $Pr_a(\phi) \geq \alpha$. The definition of the language of epistemic logic can be extended in a straightforward manner.

**Definition 1.5 (Language of probabilistic epistemic logic)**

Let a set of propositional variables $\mathcal{P}$ and a set $\mathcal{A}$ of agents be given. The language of probabilistic epistemic logic $\mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$ is the set of sentences such that:

1. $p \in \mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$ for every $p \in \mathcal{P}$
2. if $\phi, \psi \in \mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$, then $\neg \phi, (\phi \land \psi) \in \mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$
3. if $\phi \in \mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$ and $a \in \mathcal{A}$, then $K_a \phi \in \mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$
4. if $\phi \in \mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$, $a \in \mathcal{A}$ and $\alpha \in [0, 1]$, then $(Pr_a(\phi) \geq \alpha) \in \mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$
5. only those sentences that can be generated by clauses 1–4 in a finite number of steps are elements of $\mathcal{L}_{\mathcal{P},\mathcal{A}}^{pr}$.

Besides the usual abbreviations, I will use the following:

- $(Pr_a(\phi) < \alpha) \equiv_{df} \neg(Pr_a(\phi) \geq \alpha)$
- $(Pr_a(\phi) = \alpha) \equiv_{df} (Pr_a(\phi) \geq \alpha) \land (Pr_a(\neg \phi) \geq 1 - \alpha)$
- $(Pr_a(\neg \phi) \leq \alpha) \equiv_{df} (Pr_a(\neg \phi) < \alpha) \lor (Pr_a(\neg \phi) = \alpha)$
- $(Pr_a(\phi) > \alpha) \equiv_{df} \neg(Pr_a(\phi) \leq \alpha)$

$\square$
Halpern and Tuttle do not give such a definition, nor do they give a new definition of Kripke models for probabilistic epistemic logic. To keep a clear picture, I will give this definition.

**Definition 1.6 (Kripke models for probabilistic epistemic logic)**

Let a set of propositional variables $\mathcal{P}$ and a set $\mathcal{A}$ of agents be given. A Kripke model for probabilistic epistemic logic $\mathcal{M}$ is a quadruple $(W, R, V, \mathfrak{P})$ such that:

- $W$ is a nonempty set of points
- $R$ is a set of accessibility relations that contains an accessibility relation $R_a \subseteq W \times W$ for each agent $a \in \mathcal{A}$
- $V$ is a function that assigns a set of possible worlds to each proposition $p \in \mathcal{P}$
- $\mathfrak{P}$ is a probability space assignment that assigns a probability space to each agent $a \in \mathcal{A}$ at each point $c \in W$ such that its sample space is a subset of $W$.

The semantics are defined as follows. Let $S_{a,c}(\phi)$ be the subset of the sample space $S_{a,c}$ that contains all those points where $\phi$ holds. Given a model $(W, R, V, \mathfrak{P})$ such that $c \in W$:

$$c \models P_{\mathfrak{P}}(\phi) \geq \alpha \iff \mu_{a,c}(S_{a,c}(\phi)) \geq \alpha$$

A sentence $\phi$ is defined to be measurable with respect to $S_{a,c}$ if and only if $S_{a,c}(\phi) \in \mathcal{H}_{a,c}$. The only problem with the definition given above is that there is no guarantee that every sentence $\phi$ is measurable with respect to $S_{a,c}$. This can be solved by taking a slightly different definition. Instead of taking the measure, the inner measure can be taken. The inner measure $\mu_*$ of a set $A \subseteq S$ given a probability space $(S, \mathcal{H}, \mu)$ is the supremum of the measures of the subsets of $A$ that are in $\mathcal{H}$, formally:

$$\mu_*(A) = \sup \{ \mu(B) \mid B \subseteq A \text{ and } B \in \mathcal{H} \}$$

The supremum of a set of real numbers is the smallest upper bound of that set.

A number $x$ is an upper bound of a set $A$ if and only if $x$ is not smaller than any element in $A$. The other definition can now be given:

$$c \models P_{\mathfrak{P}}(\phi) \geq \alpha \iff (\mu_{a,c})_*(S_{a,c}(\phi)) \geq \alpha$$

This guarantees that the truth value of a sentence is defined. This definition gives the same results as the simpler one if a sentence is measurable with respect to $S_{a,c}$. In the analysis of the Monty Hall Dilemma it will turn out that all sentences are measurable, therefore the simpler definition will suffice.

The next question is how to choose an appropriate probability space assignment. Halpern and Tuttle show that once you have a sample space assignment $\mathfrak{S}$ that assigns a sample space $S_{a,c}$ to each agent $a$ at a point $c$, you can construct a probability space $\mathfrak{P}$. The idea is that the probability assigned to a subset $S$ of $S_{a,c}$ is the probability that a run goes through $S$ given that it goes through $S_{a,c}$. That is the conditional probability a run goes through $S$ given that it goes through $S_{a,c}$. These probabilities are calculated using the labelled
computation tree $T_A$. There are some requirements $S_{a,c}$ must satisfy for this idea to work.

**REQ1:** All points of $S_{a,c}$ are in $T(c)$.

Where $T(c)$ is the set of points that are in the same computation tree as $c$. This means all points in $a$’s sample space are contained in one computation tree. If $S_{a,c}$ does not satisfy this requirement, we would run into the same difficulties that were described in the beginning of this section, because there could be more than one adversary and therefore there could be points in one sample space from different computation trees. By limiting themselves to sample space assignment that satisfy this requirement, Halpern and Tuttle also leave any sort of meta-probability out of consideration. One could conceive of probabilities being assigned to different adversaries, somehow reflecting the chances of encountering a particular adversary. When **REQ1** is satisfied this is out of the question.

Another feature of Halpern and Tuttle’s system now becomes apparent. The probability space assignment can be independent of the information an agent has. Even if **REQ1** is satisfied it usually does not solely depend on the agent’s information. Although $P_{a}(\phi) \geq \alpha$ might hold at some point, this does not imply that $K_{a}(P_{a}(\phi) \geq \alpha)$. This is because the points that are accessible to an agent might be in different computation trees. The probability of a sentence according to the probability space that is assigned to an agent at a certain point might differ from the probability that is assigned to it by other probability spaces that are assigned to the agent at a point that is accessible to the agent.

The probability space assignment indicates which probabilities an agent should assign to certain sets of points given a certain adversary and a certain sample space assignment. If these are not encoded in the local state of the agent, then in that sense she is ignorant about the probability she should assign to sentences. Consequently $P_{a}(\phi) \geq \alpha$ does not imply $K_{a}(P_{a}(\phi) \geq \alpha)$, but $P_{a}(\phi) \geq \alpha$ is implied by $K_{a}(P_{a}(\phi) \geq \alpha)$. Halpern and Tuttle have radically separated the probability an agent assigns to a sentence from the information the agent has of that probability. In the analysis of the Monty Hall Dilemma in this chapter we only consider one computation tree, therefore this does not make any difference, but in chapter 3 this feature will be important.

A further requirement that Halpern and Tuttle casually introduce is consistency of sample spaces. Sample spaces are consistent iff $S_{a,c} \subseteq K_{a}(c)$, where $K_{a}(c) = \{ \omega' : c R_{a} \omega' \}$. This requirement implies that $K_{a} \phi \rightarrow (P_{a}(\phi) = 1)$. (If $K_{a} \phi$ is true in a point $c$, then $\phi$ holds in every point accessible to $a$ from $c$. Therefore if $S_{a,c} \subseteq K_{a}(c)$ then $\phi$ also holds in every point in $S_{a,c}$ and therefore $S_{a,c}(\phi) = S_{a,c}$. From the definition of probability spaces it follows that $\mu_{a,c}(S_{a,c}) = 1$, thus $\mu_{a,c}(S_{a,c}(\phi)) = 1$, and thus $P_{a}(\phi) = 1$.) One of the sample space assignments they introduce is however not consistent.

Since the construction of a probability space assignment from a sample space assignment involves conditional probability, another requirement must be satisfied. Let $R(S)$ be the set of runs going through the points in $S$.

**REQ2:** $R(S_{a,c}) \in H_{A}$ and $\mu_{A}(R(S_{a,c})) > 0$.

This means that the probability of a set of points $S$ can safely be divided by the probability of $S_{a,c}$. Note that **REQ2** implies **REQ1**.
1.5 Probability for Agents

Given a set of points $S_{a,c}$ assigned to an agent $a$ at a certain point $c$, the measurable subsets $\mathcal{H}_{a,c}$ comprises of exactly those subsets $H$ of $S_{a,c}$ such that the set of runs going through $H$ is a measurable subset of the probability space $(\mathcal{R}_A, \mathcal{H}_A, \mu_A)$ of the computation tree that contains $S_{a,c}$. Given a set of runs $\mathcal{R}'$ and a set of points $S$ let $Proj(\mathcal{R}', S)$ be the set of points that occur in one of the runs contained in $\mathcal{R}'$, i.e. $Proj(\mathcal{R}', S) = \{(r, k) \in S \mid r \in \mathcal{R}'\}$. The set of measurable subsets of $S_{a,c}$ can be defined as:

$$\mathcal{H}_{a,c} = \{Proj(\mathcal{R}', S_{a,c}) \mid \mathcal{R}' \in \mathcal{H}_A\}.$$ 

The probability function $\mu_{a,c}$ can now be defined as:

$$\mu_{a,c}(S) = \frac{\mu_A(\mathcal{R}(S) \mid \mathcal{R}(S_{a,c}))}{\mu_A(\mathcal{R}(S_{a,c}))}$$

Let $\mathfrak{P}_{a,c}$ be defined as $(S_{a,c}, \mathcal{H}_{a,c}, \mu_{a,c})$. Halpern and Tuttle show that if $S_{a,c}$ satisfies $REQ_1$ and $REQ_2$, then $\mathfrak{P}_{a,c}$ is a probability space. Thus given a sample space assignment $\Theta$ we can construct a probability assignment $\mathfrak{P}$, which is called the probability space assignment induced by $\Theta$.

One question still has to be answered. How do we choose a sample space assignment? Before they answer this question Halpern and Tuttle distinguish between two types of systems, synchronous and asynchronous systems. In a synchronous system all agents know what time it is. If two points $(r, k)$ and $(r', k')$ are indistinguishable to an agent in a synchronous system, then $k = k'$. Asynchronous systems do not have to satisfy this requirement. Because the Monty Hall Dilemma is a synchronous system, I will focus on these. The initial question can now be reformulated. How do we choose a sample space assignment in a synchronous system? Halpern and Tuttle restrict their attention to standard sample space assignments. Standard assignments satisfy three conditions. They are state generated, which means that for every sample space if two points have the same global state they must either both be included in it or both be excluded from it: if $(r, m) \in S_{a,c}$ and $r(m) = r'(m')$, then $(r', m') \in S_{a,c}$. (Remember that the adversary was encoded in the global state. Therefore this condition does not violate $REQ_1$, because if $r(m) = r'(m')$ then $(r, m)$ and $(r', m')$ are both in the same computation tree.) Secondly, they are inclusive, i.e. for every agent $a$ and point $c$: $c \in S_{a,c}$. Thirdly, they are uniform: if $d \in S_{a,c}$, then $S_{a,d} = S_{a,c}$. According to Halpern and Tuttle, in practice sample spaces satisfy these conditions. The great advantage of these conditions is that if a sample space assignment $\Theta$ is a consistent, standard assignment and the language $L_{P_A}$ is state generated, then every sentence $\phi$ is measurable with respect to $\Theta$. $L_{P_A}$ is state generated with respect to a system $\mathcal{R}$ if all propositional variables are facts about the global state of the system. If every sentence is measurable we do not need the inner measure truth definition.

The question how to choose a sample space assignment can still have different answers. According to Halpern and Tuttle the question can be formulated in terms of the willingness of agents to accept bets from different opponents. I do not always find this very elucidating. This is discussed more extensively in section 1.6. Therefore I will also mention some other intuitions one might have with these sample space assignments.

Four sample space assignments are discussed. The first sample space assignment is appropriate when an agent is considering accepting a bet from a copy of
himself, i.e. an opponent who has exactly the same knowledge. In this case the sample space for an agent \( a \) at point \( c \) is defined as \( S_{a,c}^{\text{post}} = \{ d \in T(c) \mid cR_a d \} \). These are all those points in the computation tree \( a \) considers possible. This sample space assignment is denoted by \( S_{a,c}^{\text{post}} \) and the induced probability space assignment by \( \mathcal{P}_{a,c}^{\text{post}} \). The superscript ‘post’ stands for the posterior probability. To most people this is intuitively the most appealing, because it comes nearest to including all points that are accessible to an agent in the sample space. It is the probability an agent should assign to the points given everything the agent knows, and the adversary.

According to Halpern and Tuttle the second probability assignment is appropriate when an agent is considering accepting a bet from an opponent who has complete knowledge of the system as it has developed up to the point at which the agent is. In this case the sample space for an agent \( a \) and a point \( c = (r, k) \) consists of those points that have the same prefix as \((r, k)\) up to time \( k \). This sample space is denoted by \( S_{a,c}^{\text{fut}} \), the sample space assignment is denoted by \( S_{a,c}^{\text{fut}} \) and the induced probability space assignment by \( \mathcal{P}_{a,c}^{\text{fut}} \). Here ‘fut’ stands for ‘future’ because only future events can have probabilities other than 0 or 1. In my view it gives a very naive perspective on probability. It can be illustrated by the following example. Suppose someone puts a die into an opaque cup and turns the cup over on the table. What is the probability that six is thrown? One might say that it is either 0 or 1, because there is a fact of the matter (putting quantum mechanical considerations aside). Only beforehand could one say that the probability of six is \( \frac{1}{6} \).

The third corresponds to accepting a bet from a different agent in the system. It turns out that the appropriate sample space assignment consists of those points in the computation tree both agents consider possible. If agent \( a \) is considering accepting a bet from agent \( b \) the sample space is defined as \( S_{a,c}^{b} = S_{a,c}^{\text{post}} \cap S_{b,c}^{\text{post}} \). This sample space is \( S_{a,c}^{b} \) and the probability space assignment is \( \mathcal{P}_{a,c}^{b} \). With this sample space assignments there are no other intuitions but willingness to bet. The previously discussed sample space assignments \( S_{a,c}^{\text{post}} \) and \( S_{a,c}^{\text{fut}} \) can be thought of as special cases of \( S_{a,c}^{b} \). Since \( S_{a,c}^{\text{post}} = S_{a,c}^{b} \cap S_{b,c}^{\text{post}} \) and if \( b \) is an agent with complete knowledge of the history of the system, then \( S_{a,c}^{\text{fut}} = S_{a,c}^{b} \cap S_{b,c}^{\text{post}} \).

The last sample space cannot be thought of in terms of bets. It simply consists of all points that are on the same level of the computation tree, i.e. given a point \( c = (r, k) \) in \( T_A \) the sample space \( S_{a,c}^{\text{prior}} = \{(r', k') \mid k' = k \text{ and } (r', k') \text{ is also in } T_A \} \). This sample space assignment is denoted by \( S_{a,c}^{\text{prior}} \) and the induced probability space assignment by \( \mathcal{P}_{a,c}^{\text{prior}} \). Here ‘prior’ means that it is the probability the agent should assign to the different events prior to any development in the system, that is at the starting point. Note that this sample space assignment is not necessarily consistent. It only depends on the computation tree the agent is in, and is completely independent of the points that are accessible to the agent.

To gain an overview of the construction of this logic a schematic representation is given in figure 1.7. Again local states form the basis of the model, but the systems are seen as computation trees. A computation tree combined with a set of adversaries brings about a set of labelled computation trees. These can be seen as probability spaces, one for each adversary, where the sample space is the set of runs of the system. Given a sample space assignment and the prob-
ability spaces associated with the adversaries, probability space assignments are constructed, where the sample spaces are sets of points. These together with the Kripke model generated by the labelled computation trees result in a probabilistic epistemic Kripke model.

Now we can use the whole system to complete the analysis of the Monty Hall Dilemma. With the labelled computation tree shown in figure 1.6 we can calculate the probability $p$ should assign to $A_1$ and $A_3$ when it is in the local state $(3,1,3)$ (the player picked door number one and the quiz master opened door number three). This was the situation described in vos Savant column. What is the most appropriate probability space assignment in this case? $P^{post}$ assigns to each agent at each point those points in the computation tree that are accessible to the agent. This corresponds to accepting a bet from a copy of itself. Because we are considering only one computation tree, the equivalence classes induced by $R_p$ are the sample spaces. In case the player is in local state $(3,1,3)$ there are two points in its sample space. Because $p$'s sample space is the same at both points we can take $c = ((3,1,1,3),3)$, then $S_{p,c}^{post} = \{(3,1,1,3),3, (3,2,1,3),3\}$. (For brevity the runs are represented by the local state of the quiz master at time 3.) The elements of the sample space are indicated by the oval nodes in figure 1.8. Because all the runs of the system are finite, the set of measurable subsets $H_{p,c}^{post}$ of a sample space $S_{p,c}^{post}$ is the powerset of $S_{p,c}^{post}$. Thus the set $S_{p,c}^{post}(A_1) = \{(3,1,1,3),3\}$ and the set
$\mathcal{S}_{p,c}^{\text{post}}(A_2) = \{(3, 2, 1, 3)\}$ are both measurable.

\[
\begin{align*}
\mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}(A_1))) &= \frac{1}{3} \cdot \frac{1}{3} \cdot \frac{1}{3} = \frac{1}{27} \\
\mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}(A_2))) &= \frac{1}{3} \cdot \frac{1}{3} \cdot 1 = \frac{1}{9} \\
\mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}})) &= \mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}(A_1))) + \mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}(A_2))) = \frac{3}{18} = \frac{1}{6} \\
\mu_{p,c}^{\text{post}}(\mathcal{S}_{p,c}^{\text{post}}(A_1)) &= \frac{\mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}(A_1)))}{\mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}))} = \frac{18}{18} = \frac{1}{6} \\
\mu_{p,c}^{\text{post}}(\mathcal{S}_{p,c}^{\text{post}}(A_2)) &= \frac{\mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}(A_2)))}{\mu_A(\mathcal{R}(\mathcal{S}_{p,c}^{\text{post}}))} = \frac{3}{18} = \frac{1}{6}
\end{align*}
\]

Therefore $c \models K_p(Pr_p(A_1) = \frac{1}{3})$ and $c \models K_p(Pr_p(A_2) = \frac{2}{9})$. This is the same result Marilyn vos Savant arrived at.

However it is still interesting to see what results we get when we take another probability space assignment. $\Psi^{\text{fast}}$ assigns to each agent $a$ at each point $c$ the sample space consisting of those points that have the same prefix as the run $a$ is in up to point $c$. This was appropriate for accepting a bet from an agent with complete knowledge of the system. $\Psi^{\text{fast}}$ determined what the appropriate probability assignment was when betting against another agent in the system. Because the only other agent in the system is the quiz master and he has complete knowledge of the system, $\Psi^{\text{fast}}$ and $\Psi^{\text{fast}}$ boil down to the same probability space assignment. The sample space assignment $\Psi^{\text{fast}}$ assigns to $p$ when it is in local state $(3, 1, 3)$ at a certain point consists of only that point. So the
oval nodes also indicate the sample spaces assigned to \( p \) at these points, except that in this case they constitute separate sample spaces. Let \( c = ((3, 1, 1, 3), 3) \). All subsets of \( S_{p,c}^{\text{fut}} \) are still measurable. Now \( S_{p,c}^{\text{fut}}(A_1) = \{ ((3, 1, 1, 3), 3) \} \), but \( S_{p,c}^{\text{fut}}(A_2) = \emptyset \).

\[
\mu_A(\mathcal{R}(S_{p,c}^{\text{fut}}(A_1))) = \frac{1}{3} \cdot \frac{1}{3} \cdot \frac{1}{2} = \frac{1}{18} \\
\mu_A(\mathcal{R}(S_{p,c}^{\text{fut}}(A_2))) = 0 \\
\mu_A(\mathcal{R}(S_{p,c}^{\text{fut}})) = \mu_A(\mathcal{R}(S_{p,c}(A_1))) = \frac{1}{18} \\
\mu_{\text{fut}}(S_{p,c}^{\text{fut}}(A_1)) = \frac{\mu_A(\mathcal{R}(S_{p,c}^{\text{fut}}(A_1)))}{\mu_A(\mathcal{R}(S_{p,c}^{\text{fut}}))} = \frac{1}{18} = 1 \\
\mu_{\text{fut}}(S_{p,c}^{\text{fut}}(A_2)) = \frac{\mu_A(\mathcal{R}(S_{p,c}^{\text{fut}}(A_2)))}{\mu_A(\mathcal{R}(S_{p,c}^{\text{fut}}))} = \frac{0}{18} = 0
\]

Therefore \( c \models K_p(Pr_p(A_1) = 1) \) and \( c \models K_p(Pr_p(A_2) = 0) \). This does not agree with Savant’s outcomes. If \( c = ((3, 3, 1, 3), 3) \) the probabilities are the other way around. Thus \( \mathcal{P}_{p,c}^{\text{fut}} \models K_p(Pr_p(A_1) = 1) \lor (Pr_p(A_2) = 1) \); \( p \) knows there is a fact of the matter, but not which.

The last probability space assignment, \( \mathcal{P}_{p,c}^{\text{prior}} \), assigns to each agent at each point all points with the same time. These are all the points at the same level of the computation tree. Let \( c = ((3, 1, 1, 3), 3) \). The sample space associated with this point is indicated by the dashed box in figure 1.6. Again \( H_{p,c}^{\text{prior}} \) is the powerset of \( S_{p,c}^{\text{prior}} \). Therefore the sets \( S_{p,c}^{\text{prior}}(A_1) = \{ ((3, 1, 1, 2), 3), ((3, 1, 1, 3), 3), ((3, 1, 2, 3), 3), ((3, 1, 3, 2), 3) \} \) and \( S_{p,c}^{\text{prior}}(A_2) = \{ ((3, 2, 1, 3), 3), ((3, 2, 2, 1), 3), ((3, 2, 3, 1), 3) \} \) are also measurable.

\[
\mu_A(\mathcal{R}(S_{p,c}^{\text{prior}}(A_1))) = \frac{1}{3} \\
\mu_A(\mathcal{R}(S_{p,c}^{\text{prior}}(A_2))) = \frac{1}{3} \\
\mu_A(\mathcal{R}(S_{p,c}^{\text{prior}})) = 1 \\
\mu_{p,c}^{\text{prior}}(S_{p,c}^{\text{prior}}(A_1)) = \frac{1}{3} \\
\mu_{p,c}^{\text{prior}}(S_{p,c}^{\text{prior}}(A_2)) = \frac{1}{3}
\]

Therefore \( c \models K_p(Pr_p(A_1) = \frac{1}{2}) \) and \( c \models K_p(Pr_p(A_2) = \frac{1}{2}) \). Although this does not agree with Savant’s outcomes, it does agree with the prior probability that \( p \) should assign to the propositions.

### 1.6 Evaluation

There are two aspects of this analysis of the Monty Hall Dilemma that are not quite satisfactory. Firstly, it is not clear how to make a choice between the various probability space assignments. Although Halpern and Tuttle say different probability space assignments are appropriate for different cases, they give no good indication how to make up your mind in a specific case. They indicate what choice to make by saying what type of bets can be associated with each of the probability space assignments. In case of the Monty Hall Dilemma one would say, intuitively, that the player is betting against the quiz master. However if you take the matching probability space assignment the
Produced results are not the same as in Savant’s analysis. This is because in their understanding of a safe bet, it must be safe against every strategy an opponent may have. So it has to be safe for the most hostile opponents, whereas in the Monty Hall Dilemma the quiz master seems to be quite generous. Yet assumptions about the generosity of the opponent cannot be taken into account. Halpern and Tuttle themselves have noted this deficiency.

The challenge that remains is to apply this new-found understanding to real problems. In order to apply our results, there is one extension of this work that might be useful. We have shown how to construct the most appropriate probability space for a given opponent in a betting game, but we have made no assumptions about the strategy the opponent is following. One potentially fruitful line of research is to understand how our results would be affected if we make assumptions about the strategies an adversary \( p_j \) is allowed to follow, such as assuming that \( p_j \) is trying to maximize its payoff and not simply trying to break even. [HT93]

Consequently, if you want to analyse a real problem with the system as it was presented in their article and you are impartial to the outcome, it is easy to make mistakes. More precise criteria for the choice of a probability space assignment should be given for this system to work as an instrument of analysis.

In my opinion \( \mathcal{S}^{\text{post}} \) is the only sample space assignment that can be taken seriously. Clearly \( \mathcal{S}^{\text{post}} \) is the sample space assignment used by Marilyn vos Savant. Her opponents do not seem to use any of the other sample space assignments to come to the conclusion that switching does not increase the probability of winning. (I will try to show in section 3.5 that they also use \( \mathcal{S}^{\text{post}} \).) I have never heard of someone using the concept of probability in the way suggested by \( \mathcal{S}^{\text{flat}} \), especially not for subjective probability. \( \mathcal{S}^{\text{prior}} \) gives a good interpretation of prior probability, however it seems very counterintuitive to give a non-zero probability to events one knows will never occur. All this seems to indicate that \( \mathcal{S}^{\text{post}} \) is the only sample space assignment that corresponds to the everyday concept of probability. But this is a pragmatic argument. I think there is no conclusive argument that can give a definitive answer to the question which sample space assignment is most appropriate. Yet in this thesis I will focus on \( \mathcal{S}^{\text{post}} \).

Secondly, in Halpern and Tuttle’s system, the reasoning about probability cannot be analysed using a formal language. For a satisfactory analysis it should be possible to formalize the arguments involved in the Monty Hall Dilemma so that you could prove them to be correct or to show with a counterexample that they are incorrect. This, however, is not possible in Halpern and Tuttle’s system. In the article by Fagin and Halpern an axiom system is presented for a comparable logic. Yet it is too weak to express the arguments involved in an analysis of the Monty Hall Dilemma. What should the set of premises be? One of the premises should say that the probability that the car is behind a door is \( \frac{1}{3} \). This could be represented by a sentence such as \( (Pr(A_1) = \frac{1}{3}) \land (Pr(A_2) = \frac{1}{3}) \land (Pr(A_3) = \frac{1}{3}) \). But from this premise one could never deduce \( (Pr(A_2) = \frac{1}{2}) \), because it contradicts the second conjunct of the premise. (It would of course be possible to deduce this if the set of premises were inconsistent.)

A reason for the failure of the formalisation, one might argue, is that temporal operators are not incorporated into the system. But these could be added.
1.6 Evaluation

Because time is discrete and linear an operator ‘\( \square \)’ meaning ‘next’ can be added. A sentence \( \square \phi \) expresses that one time point further \( \phi \) holds \((r, k) = \square \phi \) iff \((r, k + 1) \models \phi \). With this operator one could express that first a car is placed, after that the player makes a choice and then if certain conditions are satisfied one of the doors is opened. If this approach would be taken, you would need sentences such as \( Pr_p(\square A_1 \land \square \square C_1 \land \square \square \square O_2) = \frac{1}{3} \). This, in my view, would amount to ‘drawing a labelled computation tree with sentences.’ It would be an artificial formalisation. I would not consider this to be a formalisation of the argument.

I think that the most important step in an analysis that uses this system is to make a computation tree and to assign probabilities to the edges. Yet in making a labelled computation tree you are completely free. Therefore you have to analyse the problem before you can make a tree. Thus this system cannot be used as an independent instrument for analysis.
Chapter 2

Dynamic Epistemic Logic

The logic presented in this chapter stems from Jelle Gerbrandy’s dissertation *Bisimulations on Planet Kripke*. This logic is called DEL and it combines epistemic logic, which was introduced in section 1.1, with dynamic logic, which is introduced in section 2.1. Although standard semantics for both dynamic logic and epistemic logic use Kripke models, Gerbrandy uses different semantics. He uses a non-standard set theory for his approach, which is treated in section 2.2. How Kripke models and the models Gerbrandy introduces are related is discussed in section 2.3. In section 2.4 the dynamical aspects of DEL are introduced, and special attention is given to what happens in DEL when agents acquire information. Finally, in section 2.5, the non-probabilistic aspects of the Monty Hall Dilemma are analysed with DEL.

2.1 Dynamic Logic

Dynamic logic was developed to model processes as they occur inside computers. One of the most influential approaches is by Pratt [Pra76]. It is a modal logic where the internal states of a computer processing unit are taken to be the possible worlds. A computer program can bring the computer from one internal state to another, therefore the accessibility relations are based on computer programs. A computer program \( \pi \) for example can add 1 to register \( x \), that is \( x := x + 1 \). If the internal state of the processing unit is such that \( x = 7 \), then an internal state where \( x = 8 \) can be reached by executing \( \pi \). Because every program has its own accessibility relation, every program has its own modal operators. If \( \pi \) is a program, \( \langle \pi \rangle \phi \) expresses that after \( \pi \) has been executed \( \phi \) could be the case and \( [\pi] \phi \) expresses that after \( \pi \) has been executed \( \phi \) has to be the case.

Just as there are primitive or atomic propositions in propositional logic, there are atomic programs in dynamic logic. With a number of operators more complex programs can be made out of these atomic programs. One of the simplest operators is sequencing. If the program \( \pi; \pi' \) is executed, then first \( \pi \) is executed and then \( \pi' \) is executed. The accessibility relation of these programs is the composition (or relative product) of \( \pi \)'s accessibility relation and \( \pi' \)'s accessibility relation. If the program \( \pi \cup \pi' \) is executed, then either \( \pi \) is executed or \( \pi' \) is executed. This is also called nondeterministic choice. The accessibility
2.1 Dynamic Logic

relation of this operator is the union of the accessibility relations of \( \pi \) and \( \pi' \). When you are writing a program you will often want a program to be executed only if certain conditions are satisfied. There are special operators for tests. The program \( ?\phi \) will give exactly the same state where the computer was, except if \( \phi \) is not true in that state. In that case the program is aborted. This ensures that a program is only executed if certain conditions are satisfied.

With these operators you can give a formal version of a so-called IF THEN ELSE statement. An IF THEN ELSE statement is often used when a certain program should be executed if certain conditions hold, but another program should be executed if those conditions do not hold. A coffee machine for example will provide a cup and it will charge fifteen cents extra if its sensor does not detect that there is already a cup in the machine, but it will start pouring coffee if its sensor does detect a cup. A program like this could be formalized as \( (?\phi; \pi) \cup (?\neg \phi; \pi') \). In the analysis of the Monty Hall Dilemma I will often use this kind of construction.

In recent developments this approach has been extended to other areas. Communication can also be looked upon from a dynamical point of view. Natural language can be thought of as a cognitive programming language. This view is also called 'update semantics', because getting information is also called updating with information. In his influential article 'Defaults in update semantics', Frank Veltman describes this dynamic turn in logic as a change in slogans:

The slogan 'You know the meaning of a sentence if you know the conditions under which it is true' is replaced by this one: 'You know the meaning of a sentence if you know the change it brings about in the information state of anyone who accepts the news conveyed by it'. Thus, meaning becomes a dynamic notion: the meaning of a sentence is an operation on information states. [Vel96]

In this case information states play the role of possible worlds. By updating with information someone can end up in another information state. For each information state it is defined which (other) information states can be reached by updating with a certain piece of information. In this approach various notions of validity can be defined. For example: \( \psi_1, \ldots, \psi_n / \phi \) is valid iff \( \phi \) holds in any information state you end up in if you are updated with \( \psi_1, \ldots, \psi_n \) in that order. (This is Veltman's valid2 in [Vel96], where other notions of validity are also discussed.)

With update semantics you can show that the order in which information is provided can make a difference. Veltman gives a particularly good example of this:

Somebody is knocking at the door . . . Maybe it's John . . . It's Mary.
Somebody is knocking at the door . . . It's Mary . . . Maybe it's John.

[Vel96]

The first series of sentences seems quite natural, whereas the second does not.

The dynamic epistemic logic Gerbrandy has developed belongs to this tradition. His system models changes in information that involve more than one agent too. What happens for example if one agent is updated, but another one is not and what happens if a group of agents is updated simultaneously?
2.2 Non-well-founded Sets

Gerbrandy uses non-well-founded set theory for the semantics of epistemic logic and dynamic epistemic logic. This set theory was developed by Aczel [Acz88]. It is a non-standard set theory in which sets can be an element of themselves and functions can apply to themselves as an argument or yield themselves as a value. The reason for using this set theory is that it has several technical advantages. There are also some philosophical considerations that play a role. These will become apparent in the following sections.

Barwise and Moss have written a very accessible book about non-well-founded set theory called Vicious Circles [BM96]. They argue that if a phenomenon is somehow circular and cannot be modeled in standard set theory, this is no reason to despair.

In certain circles, it has been thought that there is a conflict between circular phenomena, on the one hand, and mathematical rigor, on the other. This belief rests on two assumptions. One is that anything mathematically rigorous must be reducible to set theory. The other assumption is that the only coherent conception of set precludes circularity. As a result of these two assumptions, it is not uncommon to hear circular analyses of philosophical, linguistic, or computational phenomena attacked on the grounds that they conflict with one of the basic axioms of mathematics. But both assumptions are mistaken and the attack is groundless. [BM96, p.5]

The simplest examples they give of non-well-founded phenomena are so-called streams [BM96, p.3 and p.4].

\[
\text{week} = (\{\text{Su}, \{\text{Tu}, \{\text{W}, \{\text{T}, \{\text{H}, \{\text{F}, \{\text{Sat}, \text{week}\}\}\}\}\}\}\}\}
\]

\[
\text{conversation} = (1^{\text{st}} \text{ speaker}, (2^{\text{nd}} \text{ speaker}, \text{ conversation}))
\]

These can be seen as characterizations of time and two person conversations (quite long ones). If you take these sort of definitions seriously, you end up with non-well-founded set theory.

The best way to explain the idea of non-well-founded set theory is to view sets as graphs. A graph is defined as a set of nodes and a set of edges. An edge is an ordered pair of nodes. If \((n_i, n_j)\) is an edge this can be represented as \(n_i \rightarrow n_j\); the node \(n_j\) is also called a child of \(n_i\). We are especially interested in so-called accessible pointed graphs (apgs), because these can represent sets. A pointed graph is a graph with a distinguished node called its point. A node \(n_j\) is accessible from a node \(n_i\) if \(n_i = n_j\) or if there is a sequence of edges (also called a path) \(n_i \rightarrow \cdots \rightarrow n_j\) connecting the two nodes. If every node is accessible from the point of a pointed graph the pointed graph is an accessible pointed graph.

One of the simplest examples that illustrates this idea is to view the natural numbers as apgs. A natural number can be defined as the set containing all its predecessors; that is \(0 = \emptyset\), \(1 = \{0\} = \{\emptyset\}\), \(2 = \{0, 1\} = \{\emptyset, \{\emptyset\}\}\), etc. The apgs are shown in figure 2.1. A set is represented as a node, its elements are its children. This can be made more precise. A decoration \(\delta\) of a graph is a function that assigns a set to each node such that it comprises the sets that are assigned to its children as its elements. \(\delta(n) = \{\delta(n') : n \rightarrow n'\}\). If a decoration of the point of an apg is a set, the apg is called a picture of that set.
2.2 Non-well-founded Sets

A well-founded graph does not contain any infinite paths $n_0 \rightarrow n_1 \rightarrow n_2 \rightarrow \cdots$. A number of interesting properties are displayed by well-founded graphs. They all have unique decorations and every well-founded set has a picture. Of course there may be more than one picture of the same set. For example the number 3 can also be represented as a different apg as is shown in figure 2.2. A canonical picture of a set $a$ contains those nodes that appear in sequences $a_0, a_1, a_2, \ldots$ such that $\ldots \in a_2 \in a_1 \in a_0 = a$. That means there are no superfluous nodes. Node $a_0$ is taken as the point. The set of edges contains those pairs $(x, y)$ such that $y \in x$. Any apg can be ‘unfolded’ into a tree; for instance the number 3 can be unfolded into the picture shown in figure 2.2. This is done by taking the finite paths from the point in the apg as the nodes and taking those pairs $(x, y)$ such that $x$ has length $i$ and $y$ has length $i + 1$ and $x$ is the same up to $i$ as the set of edges. The point is the path $a_0$ of length one, which is also called the root of the tree. The unfolded tree of a canonical picture of a set is called the canonical tree picture of that set. Figure 2.2 is the canonical tree picture of the number 3.

A non-well-founded set can contain infinite paths $n_0 \rightarrow n_1 \rightarrow n_2 \rightarrow \cdots$. This is the difference between well-founded set theory and non-well-founded set theory. Aczel's anti-foundation axiom says that every graph has a unique decoration, regardless of its well-foundedness. The simplest example of a non-well-founded apg is shown in figure 2.3. This set is an element of itself and has no further elements. It could be defined by the equation $\Omega = \{ \Omega \}$. These sets also have canonical pictures and can be unfolded into canonical tree pictures.
This is also shown in figure 2.3. (The fan shape functions as ellipsis.)

An important question is what non-well-founded sets add to set theory. What is gained when your conceptual tolerance regarding sets is broadened to include non-well-founded sets? The development of non-well-founded set theory was mainly motivated by its applications in computer science and epistemic logic. Therefore it seems better to refrain from asking these questions until at least one application has been studied.

2.3 Possibilities and Kripke Models

The models for epistemic logic that Gerbrandy uses are certain non-well-founded structures called possibilities. (This term is related to possible worlds in Kripke semantics and it has nothing to do with probability theory.) Possibilities are defined provocatively as follows:

**Definition 2.1 (Possibilities)**

Let $A$, a set of agents, and $P$, a set of propositional variables, be given. The class of possibilities is the largest class such that:

- A possibility $w$ is a function that assigns to each propositional variable $p \in P$ a truth value $w(p) \in \{0, 1\}$ and to each agent $a \in A$ an information state $w(a)$.

- An information state is a set of possibilities. [Ger98, p. 12]

It is clear that this definition is circular. Possibilities are defined in terms of information states and information states are defined in terms of possibilities. However it turns out that there are non-well-founded structures that comply with this definition. The following equations for example define two possibilities, i.e. $w$ and $v$.

$$
\begin{align*}
  w(p) & = 1 \\
  w(a) & = \{w, v\} \\
  w(b) & = \{w, v\} \\
  v(p) & = 0 \\
  v(a) & = \{w, v\} \\
  v(b) & = \{w, v\}
\end{align*}
$$

One can also make pictures of possibilities in the way that was introduced in the previous section. There is however a more practical way to picture possibilities.
2.3 Possibilities and Kripke Models

This is done with labelled graphs. A labelled graph is a tuple \((G, (-\rightarrow_a)_{a \in \mathcal{A}}, V)\), where \(G\) is a set of nodes, \(\rightarrow_a\) is a set of edges for each agent \(a \in \mathcal{A}\), and \(V\) is a function that assigns to each \(p \in \mathcal{P}\) a subset of \(G\). The labelled graph of the possibility in the example above is shown in figure 2.4. An open node indicates that \(p\) is false, and a filled node indicates that \(p\) is true in that node. An unfolded tree picture is also shown in figure 2.4, where the point is \(w\). In the tree picture the normal arrows indicate \(a\)'s accessibility relation and the dashed lines indicate \(b\)'s accessibility relation. A decoration of a labelled graph is defined as a function \(\delta\) that assigns a function to each node \(w\) and each agent \(a \in \mathcal{A}\) such that:

- \(\delta(w)(p) = 1\) iff \(w \in V(p)\)
- \(\delta(w)(a) = \{\delta(w') | w \xrightarrow{a} w'\}\). [Ger98, pp.14–15]

When you examine the definition of labelled graphs closely, you will note that labelled graphs are simply multi-modal Kripke models. There is therefore a close connection between possibilities and Kripke models. Each possibility has a Kripke model as its picture and each Kripke model has a unique decoration which is a possibility.

As was the case with normal sets, different labelled graphs can be the picture of the same possibility. What is interesting is that the class of Kripke models that picture one possibility is a bisimulation class. To be more precise: a bisimulation class of pointed Kripke models. A pointed Kripke model is defined to be a Kripke model with a distinguished world; a pointed Kripke model is a pair \((M, w)\) such that \(M = (W, (-\rightarrow_a)_{a \in \mathcal{A}}, V)\) and \(w \in W\). Bisimulation is defined as follows.

**Definition 2.2 (Bisimulation)**

A relation \(R \subseteq W \times W'\) is a bisimulation between two models \(M = (W, (-\rightarrow_a)_{a \in \mathcal{A}}, V)\) and \(M' = (W', (-\rightarrow'_a)_{a \in \mathcal{A}}, V')\) iff for all \(w \in W\) and \(w' \in W'\), if \(wRw'\) then:

1. \(w \in V(p)\) iff \(w' \in V'(p)\) for all \(p \in \mathcal{P}\).
2. For all \(v\) such that \(w \xrightarrow{a} v\), there is a \(v'\) such that \(w' \xrightarrow{a'} v'\) and \(vRv'\).
3. For all \( v' \) such that \( w' \xrightarrow{a} v' \), there is a \( v' \) such that \( w \xrightarrow{a} v \) and \( v \text{Re}^l \).

Two Kripke models \( (M,w) \) and \( (M',w') \) are bisimilar, \( (M,w) \simeq (M',w') \), iff there is a bisimulation \( R \) between \( M \) and \( M' \) connecting \( w \) with \( w' \).

A bisimulation class is a class of Kripke models that are bisimilar.

A truth definition can be given for epistemic logic using possibilities.

**Definition 2.3 (Truth in possibilities)**

Let \( w \) be a possibility.

\[
\begin{align*}
    w & \models p \quad \text{iff} \quad w(p) = 1 \\
    w & \models \phi \land \psi \quad \text{iff} \quad w \models \phi \text{ and } w \models \psi \\
    w & \models \neg \phi \quad \text{iff} \quad w \not\models \phi \\
    w & \models K_a \phi \quad \text{iff} \quad \forall v \in w(a) : v \models \phi \quad \text{[Ger98, pp. 37–38]} \quad \square
\end{align*}
\]

A property of two bisimilar pointed Kripke models \( (M,w) \) and \( (M',w') \) is that they satisfy the same sentences. Therefore, though Kripke models are more finely graded than possibilities, this does not show in the semantics, because bisimilar Kripke models cannot be distinguished semantically. For Gerbrandy this is a reason to prefer possibilities to Kripke models.

We saw [...] that different models for epistemic logic (Kripke models, possibilities and knowledge structures) are all equivalent “modulo bisimulation.” Also, classical (infinitary) epistemic logic cannot distinguish between bisimilar models. The differences between bisimilar Kripke models can be, and usually are, ignored in a lot of the formal work on epistemic semantics: in completeness proofs, for example, models are being unraveled and quotients are taken of them without any further discussion. All this is circumstantial evidence for the correctness of ignoring distinctions between bisimilar Kripke models and using possibilities as our models instead. [Ger98, p.39]

Possibilities contain precisely what is relevant for epistemic logic. Still, it is often easier to understand a Kripke model than a set of equations defining a possibility. What is confusing about the relation between possibilities and Kripke models is that when you are considering a possibility you cannot make a clear distinction between possible worlds and Kripke models, because a possibility is both of these rolled into one. A decoration function assigns a certain possibility to a node in a Kripke model, therefore this node could be regarded as a possibility. However, which possibility this is, depends on the rest of the Kripke model, therefore the whole Kripke model should be regarded as a possibility. The Kripke models in figure 2.4 can be regarded in both of these ways. Often I will not make a sharp distinction between possibilities, possible worlds, Kripke models and pointed Kripke models. But, although there is a close connection between all of these, one should always be aware that there is a difference. Different possible worlds in a Kripke model can have the same decoration and if such a pointed Kripke model is taken to be the picture of a possibility, that means that different nodes can represent the same possibility.

\[\text{[Ger98, p.10] a similar definition is given for Kripke models with only one accessibility relation.}\]
2.4 Updates

Gerbrandy combines epistemic logic (see section 1.1) with dynamic logic (see section 2.1). In this system you can reason about knowledge and about information change. As was shown in the previous section, a semantics for epistemic logic can be defined in terms of possibilities. Possibilities must be seen as models for information in general and not so much as a model for knowledge. The operator $K_a$ will however still be used. The question Gerbrandy tries to answer is what happens to the information states if agents believe that the information they get is true, regardless of whether the information is actually true or not. Before I introduce the technical machinery to reason about information change I will discuss one of Gerbrandy’s examples.

Suppose there are two agents, $a$ and $b$. Both of them do not know whether a certain proposition $p$ holds. It is common knowledge that they are ignorant regarding $p$. This situation can be modeled using possibilities.

$$
\begin{align*}
  w(p) &= 1 \\
  w(a) &= w(b) = \{w, v\} \\
  v(p) &= 0 \\
  v(a) &= v(b) = \{w, v\}
\end{align*}
$$

A picture of these possibilities was shown in figure 2.4 (page 31). What happens if $a$ learns that $p$ is true? First of all the truth value of $p$ stays the same. For $b$ nothing changes, because $a$ is the only one getting information. But $a$’s situation changes; after she has learned that $p$ is true, she knows $p$ is true. The possibility $w_1$, where $a$ ends up in when she has learnt that $p$, is defined as follows:

$$
\begin{align*}
  w_1(p) &= w(p) \\
  w_1(b) &= w(b) \\
  w_1(a) &= \{w_1\}
\end{align*}
$$

A picture of this is shown in figure 2.5. In $w_1$ $a$ knows $p$ is the case, yet $b$’s information state has not changed. The result is that $b$’s information is such
that he does not consider the actual world possible. The other way to model
a's information change would be to take the Kripke model from figure 2.4 and
simply cross out the a above the arrows connecting the two worlds. As a result
however b's information would also change, because in that case b would know
that a knows the truth value of p. It seems counterintuitive that b's information
state would change if he does not get any information. One could argue that if
b were smarter she should have considered it possible that a had learned that
p. Gerbrandy prefers not to change the information of agents who do not get
information. He also takes the updates to be conscious. That means that if you
know something after you have been updated, you also know that you know it.
The operation on the canonical tree picture is very simple. All those possi-
bilities in a's information state at the root of the tree that do not agree with the
information a was updated with, are removed from a's information state. The
same operation is performed on the remaining possibilities in a's information
state.

To be able to handle information change formally Gerbrandy introduces a
notion of programs taken from dynamic logic.

To express change, I will introduce 'programs' in the object language.
These programs will be interpreted as relations between possibilities:
if π is a program, then its interpretation [π] will be that relation
between possibilities that holds between two possibilities w and v
just in case v is a possible output of executing π in w. [Ger98, p.88]

However the programs that are used in modeling computer processes are not
enough. Gerbrandy's system contains an extra type of program to model informa-
tion change.

The most important of these is written as U_a: if π is a program, then
U_aπ expresses that a consciously learns that π has been successfully
executed; I will often say that a updates with π. [Ger98, pp.88 - 89]

A formal language can now be defined that contains both epistemic operators
and programs.

Definition 2.4 (Language of DEL)
Given a set of agents A and a set of propositional variables \( \mathcal{P} \), we define the
sentences and programs of DEL simultaneously as follows.

The set of sentences of DEL is the smallest set that contains \( \mathcal{P} \), and such
that if \( \phi \) and \( \psi \) are sentences and π is a program and a is an agent, then \( \neg \phi \),
(\( \phi \land \psi \)), \( K_a \phi \) and [π]φ are sentences of DEL.

The set of programs is the smallest set that contains ?φ for each sentence φ,
and for which it holds that if π and π' are programs and a an agent, then \( U_a \pi \),
(π; π') and (π ⊔ π') are programs as well.\(^2\)

The usual abbreviations are used. The truth definition (definition 2.3, page 32)
has to be extended in such a way that it is able to handle programs. This can
be done by adding the following clause.

\[ w \models [\pi] \phi \iff \text{for all } v \text{ if } w[\pi]v \text{ then } v \models \phi \]

\(^2\)This definition is taken from [Ger98, pp.89 - 90]. It differs in a few details. In the
definition of sentences 'and a is an agent' was added and the parentheses were added in the
definition of sentences and programs.
The relation $[\pi]$ holds between two possibilities $w$ and $v$ iff $v$ is a possibility that can be reached by executing $\pi$ in $w$. The most interesting relation is the one for update programs $U_\alpha \pi$.

Programs of the form $U_\alpha \pi$ are to be read as “$\alpha$ learns that $\pi$ has been successfully executed,” or, alternatively, as “$\alpha$ updates her information state with $\pi$.” This is modeled as follows. Executing a program of the form $U_\alpha \pi$ in a possibility $w$ results in a new possibility $v$ in which only $\alpha$’s information state has changed. The information state of $a$ in $v$ contains all and only those possibilities that are the possible result of a successful execution of $\pi$ in one of the possibilities in $a$’s information state in $w$. Since we want updates to be conscious, we also change the resulting possibilities to the effect that $a$ has updated with $\pi$. So, the state of $a$ after an update with $U_\alpha \pi$ contains exactly those possibilities $v'$ that are the result of updating a possibility $w'$ from the old state with $\pi$, and then with $U_\alpha \pi$. [Ger98, p. 91]

The different accessibility relations are defined as follows. In this definition $w[a]v$ is an abbreviation for ‘$w$ and $v$ differ at most in the information state they assign to $a$.’

\[
\begin{align*}
    w[\emptyset]v & \text{ iff } w \models \emptyset \text{ and } w = v \\
    w[U_\alpha \pi]v & \text{ iff } w[a]v \text{ and } v(a) = \{a' \mid \exists w' \in w(a) \exists u : w'[\pi][u][U_\alpha \pi][v'] \} \\
    w[\pi; \pi']v & \text{ iff } \exists u \text{ such that } w[\pi][u][\pi'][v] \\
    w[\pi \cup \pi']v & \text{ iff } \text{ either } w[\pi]v \text{ or } w[\pi']v [\text{Ger98, p. 90}] \\
\end{align*}
\]

(The definition of $U_\alpha \pi$ again is circular.) Another program can be added to this list. It is one of the most interesting, the ‘group update.’ When a group update occurs a whole group of agents gets information, and all the members of the group are aware of the fact that the whole group is getting the information. The definition of its accessibility relation is analogous to the definition of an update concerning only one agent. Analogous to $w[a]v$, $w[\mathcal{B}]v$, where $\mathcal{B} \subseteq \mathcal{A}$, is an abbreviation for ‘$w$ and $v$ differ at most in the information state they assign to members of $\mathcal{B}$.’

**Definition 2.5 (Group updates)**

For each $\pi$ and $\mathcal{B} \subseteq \mathcal{A}$:

\[
    w[U_\mathcal{B} \pi]v \text{ iff } w[\mathcal{B}]v \text{ and } \forall a \in \mathcal{B} : \\
    v(a) = \{a' \mid \exists w' \in w(a) \exists u : w'[\pi][u][U_\mathcal{B} \pi][v'] \} \quad \Box
\]

One could view an update of a single agent as a special case of a group update. Group updates are also conscious. The new information states of the members of $\mathcal{B}$ do not simply contain the possibilities that were in the old information states and were accessible by executing $\pi$, but the information states of the agents in these possibilities are also updated. Otherwise they might know that $\pi$ has been successfully executed, but not know that they know it has been successfully executed.

Besides semantics Gerbrandy also provides a sound and complete axiomatization of DEL. For some this axiomatization might be more easily understood than the semantics.

\footnote{This definition is taken from [Ger98, p. 92]; but $\exists u$ and $u$ were inserted to make the definition completely analogous to the definition of a single agent update.}
Axioms

1 \( \vdash \phi \), if \( \phi \) is valid in classical propositional logic

2 \( \vdash K_\alpha(\phi \rightarrow \psi) \rightarrow (K_\alpha \phi \rightarrow K_\alpha \psi) \)

3 \( \vdash [\pi](\phi \rightarrow \psi) \rightarrow ([\pi] \phi \rightarrow [\pi] \psi) \)

4 \( \vdash [?\phi] \psi \leftrightarrow (\phi \rightarrow \psi) \)

5 \( \vdash \neg [U_B \pi] \psi \leftrightarrow [U_B \pi] \neg \psi \) (functionality)

6 \( \vdash [U_B \pi] p \leftrightarrow p \)

7 \( \vdash [U_B \pi] K_\alpha \psi \leftrightarrow K_\alpha [\pi] [U_B \pi] \psi \) if \( \alpha \in B \) (Ramsey Axiom)

8 \( \vdash [U_B \pi] K_\alpha \psi \leftrightarrow K_\alpha \psi \) if \( \alpha \notin B \) (privacy)

9 \( \vdash [\pi; \pi'] \psi \leftrightarrow [\pi] [\pi'] \psi \)

10 \( \vdash [\pi \cup \pi'] \psi \leftrightarrow (\pi \psi \land [\pi'] \psi) \)

Rules

MP \( \phi, \phi \rightarrow \psi \vdash \psi \)

NecK If \( \vdash \phi \) then \( \vdash K_\alpha \phi \)

We write \( \Gamma \vdash \phi \) or \( \Gamma \vdash_{DELK} \phi \) just in case there is a derivation of \( \phi \) from the premises in \( \Gamma \) using the rules and axioms above. The logics DELK45 and DELS5 are obtained from DEL by adding the extra axioms of K45 and S5 respectively, and we use the notation \( \vdash_{DELK45} \) and \( \vdash_{DEL55} \) accordingly. [Ger98, pp. 93–94]

I will expound briefly on the axioms in which the update operator occurs. Axiom number 5, from right to left, expresses that it is not the case that after an update \( \psi \) always holds if \( \neg \psi \) always holds after that update. This means that an update can always be executed. The relation \([U_B \pi] \) is serial. (In alethic modal logic this would mean that axiom D holds \((\Box \phi \rightarrow \Diamond \phi)\). From left to right axiom 5 expresses that if it is not the case that \( \psi \) always holds after an update, \( \neg \psi \) will always hold (i.e. \( \psi \) will never hold.) That means that the update operator is functional. There is exactly one possibility you can end up in if, in a certain possibility, the update is executed. That means there is no difference between \([U_B \pi] \phi \) and \( \neg [U_B \pi] \neg \phi \). (In the alethic modal logic, this would mean that there is no difference between necessity (\( \Box \)) and possibility (\( \Diamond \)).)

Axiom number 6 expresses that the information you get does not affect what is the case on a propositional level. Suppose for example that it is raining. If you get the information that you have won a lottery this does not change anything about the rain. Even if you would get the information that the sun is shining, it would still be raining, because the information would be false.

Axiom number 7 is more difficult. Gerbrandy elaborates on this axiom.

The Ramsey axiom, axiom number 7, expresses that after a group update with \( \pi \), an agent in the group knows \( \psi \) just in case he already knew that after executing \( \pi \), an update with \( U_B \) could only result in a possibility where \( \psi \) is true. [Ger98, p.94]
Suppose for example that after I have learned it is raining I know I will take my umbrella if I go out. According to the axiom this is the case if I know that if it is raining and I learn about it, then afterwards I will take my umbrella if I go out. (For a more extensive discussion of the background of this axiom see [Ger98, p. 102].)

Axiom number 8 says that if a group is updated and you are not a member of the group, this does not affect your information state. Suppose that everyone who watched the news, got the information that there will be a railway strike tomorrow. If you did not watch the news and did not know it by other means, you will not know about this.

It is interesting to note that the axiom system for DEL can also be extended with axioms A3, A4 and A5. In section 1.1 I indicated that if these axioms hold in a model, then the accessibility relations have certain properties. Gerbrandy gives some interesting characterizations of the classes of possibilities that satisfy these axioms.

**Definition 2.6 (Introspection and factivity)**

A class of possibilities $S$ is closed iff it holds that if $w \in S$ and $v \in w(a)$ then $v \in S$.

1. The class of positively introspective possibilities $\mathcal{P}$ is the largest closed class such that for each $w \in \mathcal{P}$ it holds that $v \in w(a)$ implies that $v(a) \subseteq w(a)$

2. The class of negatively introspective possibilities $\mathcal{N}$ is the largest closed class such that for each $w \in \mathcal{N}$ it holds that $v \in w(a)$ implies that $w(a) \subseteq v(a)$

3. The class $\mathcal{I}$ of fully introspective possibilities is the (closed) class $\mathcal{P} \cap \mathcal{N}$.

4. The class of reflective possibilities $\mathcal{T}$ is the largest closed class such that $w \in \mathcal{T}$ implies $w \in w(a)$ [Ger98, p.41]

A peculiarity of DEL is however that by executing an update in a factive possibility can result in a possibility that is not factive. The example given in section 2.4 gave an example of this. The possibility in figure 2.4 (page 31) is factive, but the possibility in figure 2.5 is not.

DEL is not a theory that is easily understood. Yet it provides an excellent model for reasoning about knowledge and information change. With DEL the muddy children problem and the hangman paradox for example can be analysed very well. Moreover these analyses seem to correspond with the way we reason about knowledge and information change in the real world.

One question still remains unanswered. Do we really need non-well-founded set theory to make the system work? The most important reason for using possibilities instead of Kripke models was that different Kripke models are the same with respect to all the sentences they satisfy, whereas possibilities are unique in that respect. This seems to be an argument for simplicity. Do not use structures you do not need! However for some people non-well-founded set theory tries to expand the universe of sets beyond its limits. They would discard this theory with Ockham’s razor.

To me it seems that canonical tree pictures are very suitable for a dynamic epistemic logic and the dynamic operators could well be defined on them. Infinite trees can be treated very well using standard set theory. Yet I will not
try to back up the claim that we can do without non-well-founded set theory, because it lies outside the scope of this thesis. DEL is a system that works quite well at modeling reasoning about knowledge and information change and that is all I am interested in at the moment.

2.5 An Analysis of the Monty Hall Dilemma

The most obvious problem of an analysis of the Monty Hall Dilemma with DEL is that probability is not incorporated into the system. That means that sentences like ‘according to agent α, φ has at least probability α’ cannot be formalized. However, much of the information the quiz master and the player acquire have nothing to do with probability. The quiz master, for example, learns behind which door the car is placed. These changes in information can be analysed quite satisfactorily in a system without probability. Moreover such an analysis can help in formulating how probabilities should change after an update.

Yet there is another difficulty that impedes a satisfactory analysis of these changes in information. In DEL the information the agents have may change, but the world never changes in such a way that the truth value of a sentence without epistemic or dynamic operators changes. The agents cannot go from one possibility to another in such a way that the truth value of a propositional variables changes. In the Monty Hall Dilemma sentences do change in truth value. The player makes a choice and a door is opened. Consequently the truth value of the sentence ‘a door is open,’ for instance, changes. This cannot be modeled in DEL.

There is a way to deal with this difficulty, but it is not very elegant. One can say there already is a fact of the matter. Ultimately a car is behind one of the doors, a choice has been made and one door is open, but neither the quiz master nor the player are aware of this. Thus they learn about their own actions as the quiz progresses. From a deterministic point of view this would not be a problem. However one would rather not have to make a choice between determinism and nondeterminism in the system, so as to make it as general as possible. But for now it will have to do. I think (non)determinism is not essential for the Monty Hall Dilemma.

The idea of the following analysis is to show what happens to both the player’s and the quiz master’s information. One can try to follow the informal description of the problem as it was put to Marilyn vos Savant as closely as possible. This description begins as follows ‘Suppose you’re on a game show, and you’re given the choice of three doors. Behind one door is a car, behind the others goats.’ [vS90] If one would follow this precisely one would first have to see what happens if the agents get the information that the player may only choose one door. After that one would have to see what happens if the agents get the information that there is one car behind one of the doors and goats behind the others, and so on. This would model what happens to the information of someone who reads the problem. But that is not what I set out to do. I am interested in what it is like for the player and the quiz master. To do this I will use the following strategy. First I analyse what happens to the information of the player and the quiz master ‘before they go on stage.’ That is, they learn the rules of the quiz. According to the rules there is exactly one car behind the doors and goats behind the others, the player may make one choice and the
2.5 An Analysis of the Monty Hall Dilemma

The quiz master may open exactly one door. The quiz master cannot just open any door, there are a number of conditions that must be satisfied. Both the player and the quiz master have to learn these conditions. After they have learned the rules of the quiz they are ready to go on stage. Just before they go, the quiz master learns behind which door there is a car and behind which doors there are goats. The player and the quiz master both have to learn by a conscious group update that the quiz master has learned this, so as to ensure it is common knowledge that the quiz master has learned where the car is. Then they go on stage. The player may choose one of the doors. As I explained above the way in which I will analyse this is to let the player and the quiz master both learn which door is chosen. Finally a door is opened by the quiz master. Again both the player and the quiz master learn which door is opened. Then the player has to decide whether he wants to switch doors. I will not be able to say in this chapter whether it would be wise for the player to profit from the quiz master’s offer. Here only the non-probabilistic information changes will be modeled.

We can formalize the nonprobabilistic updates adequately in DEL. We can take the same set of agents and propositional variables as in chapter 1. Thus $\mathcal{A} = \{p, q\}$ and $\mathcal{P} = \{A_1, A_2, A_3, C_1, C_2, C_3, O_1, O_2, O_3\}$. Let us assume that no one knows anything about these sentences. That means that for both agents at least 512 ($= 2^9$) possibilities are accessible. I will assume we are dealing with an $S_5$ scenario and that the ignorance of both agents is common knowledge. First of all the agents get the information that there is only one car behind the doors (and two goats behind the others). The sentence onecar expresses this.

$$\text{onecar} = \bigvee_{1 \leq i \leq 3} (A_i \land \bigwedge_{j \neq i, 1 \leq j \leq 3} \neg A_j)$$

Moreover the player chooses only one door. This is expressed by onechoice.

$$\text{onechoice} = \bigvee_{1 \leq i \leq 3} (C_i \land \bigwedge_{j \neq i, 1 \leq j \leq 3} \neg C_j)$$

Furthermore only one door may be opened, which is expressed by oneopen.

$$\text{oneopen} = \bigvee_{1 \leq i \leq 3} (O_i \land \bigwedge_{j \neq i, 1 \leq j \leq 3} \neg O_j)$$

After they have learned this, that is after $U_{(a, b)}(\text{onecar} \land \text{onechoice} \land \text{oneopen})$ has been executed, the situation can be modeled as a three dimensional Kripke model, as is shown in figure 2.6. The arrows indicate the accessibility for both agents. In this picture the reflexive arrows and transitive arrows are not shown. In a complete picture any two nodes would be connected by both accessibility relations. If one really views this Kripke model as a picture of a possibility it should be a pointed Kripke model. It does not really matter which of the worlds one regards to be the point. Let us say it is the world where the car is behind door number two, the player chooses door number one and door number three is opened. As will become apparent later on, this assumption does not affect the generality of the analysis.

Now the player and the quiz master learn the conditions for opening a door. Both $p$ and $q$ are updated with this information, because the quiz master also
has to know these conditions. They can be formulated as follows.

\[
\text{conditions } = \text{qlearnsA;} U_{\{p,q\}} \text{qlearnsA;} \text{pchooses}
\]

\[
\begin{align*}
(O_1 & \leftrightarrow (K_q \neg A_1 \wedge \neg C_1 \wedge \neg O_2 \wedge \neg O_3)) \wedge \\
(O_2 & \leftrightarrow (K_q \neg A_2 \wedge \neg C_2 \wedge \neg O_1 \wedge \neg O_3)) \wedge \\
(O_3 & \leftrightarrow (K_q \neg A_3 \wedge \neg C_3 \wedge \neg O_1 \wedge \neg O_2))
\end{align*}
\]

I have used some undefined programs for this definition, viz. \text{qlearnsA} and \text{pchooses}. Intuitively these mean, as their names suggest, that \( q \) learns where the car is and that \( p \) chooses a door. These will be defined below. The sentence \text{conditions} expresses that after the quiz master has learned where the car is, the player and the quiz master have both learned that the quiz master has learned this, and the player has picked a door, then the quiz master may open a door if and only if he knows it does not contain the car, the player did not pick that door and the other doors are not opened.

The player and the quiz master learn these conditions, that is the program \( U_{\{p,q\}} \text{?conditions} \) is executed. Now our task is to find out what happens to the information states of the two agents. If we take as the point the world where the car is behind door number two, the player picks door number one and door number three is opened, we must scrutinize the possibilities in the information states of the agents and see in which of these \text{conditions} hold. The new information states of the agents will contain those possibilities where \text{conditions} hold, after they have been updated with the test \text{?conditions} (that is, after the program \( U_{\{p,q\}} \text{?conditions} \) has been executed in those remaining possibilities, so as to make the update conscious). Consequently we have to find out in which of the 27 possibilities \text{conditions} actually holds. Therefore we have to find out which possibility is reached in each of these possibilities by executing...
2.5 An Analysis of the Monty Hall Dilemma

\[ q \text{learnsA} : U_{[p,q]} q \text{learnsA}, p \text{chooses} \]. This is just like checking whether a sentence of the form \( \square \phi \) holds in an alethic modal model. We have to check whether \( \phi \) holds in those worlds that can be reached from a given world with three arrows. In case of conditions however the possible worlds are possibilities and the arrows are accessibility relations associated with the programs \( q \text{learnsA}, U_{[p,q]} q \text{learnsA} \), and \( p \text{chooses} \).

What happens if the quiz master learns behind which door there is a car and behind which doors are goats. It means the following program is executed:

\[
q \text{learnsA} = (\langle ?A_1; U_q ?A_1 \rangle \cup (\langle ?\neg A_1; U_q \neg A_1 \rangle);
\langle ?A_2; U_q ?A_2 \rangle \cup (\langle ?\neg A_2; U_q \neg A_2 \rangle);
\langle ?A_3; U_q ?A_3 \rangle \cup (\langle ?\neg A_3; U_q \neg A_3 \rangle)
\]

These are three IF THEN ELSE statements that are executed in sequence. If \( A_1 \) is the case \( q \) will be updated with \( A_1 \), otherwise \( q \) will be updated with \( \neg A_1 \). The same happens with \( A_2 \) and \( A_3 \).

In a simpler case the effect of this update can be better understood. What if there were only three possibilities, one for each proposition \( A_1 \), \( A_2 \) and \( A_3 \). Let these possibilities be \( u, v \) and \( w \):

\[
\begin{align*}
u(A_1) &= 1 \\
u(A_2) &= u(A_3) = 0 \\
v(A_2) &= 1 \\
v(A_1) &= v(A_3) = 0 \\
w(A_3) &= 1 \\
w(A_1) &= w(A_2) = 0 \\
u(p) &= v(p) = w(p) = \{u, v, w\} \\
u(q) &= v(q) = w(q) = \{u, v, w\}
\end{align*}
\]

We assumed that in the actual situation is \( A_2 \) is the case. This possibility is shown as a tree in figure 2.7. To keep the picture clear I have used the solid round dot to indicate a node that a decoration function would assign the possibility \( u \).
to. An x shaped node indicates node that would be decorated with v, and an
open square node indicates w would be assigned to that node. A normal arrow
indicates p's accessibility and a dashed arrow indicates q's accessibility. The
accessibility relations are separated because this will make it easier to perform
an update, as was shown in section 2.4. Now the program qleamsA is executed.
What happens is completely analogous to the first example that was discussed
in section 2.4. The information p has does not change, but the information q
has does change. The new possibility \( \nu_1 \) is defined by:

\[
\begin{align*}
\nu_1(A_2) &= 1 \\
\nu_1(A_1) &= \nu_1(A_3) = 0 \\
\nu_1(p) &= \{u, v, w\} \\
\nu_1(q) &= \{v_1\}
\end{align*}
\]

The new picture is shown in figure 2.5. The possibility \( \nu_1 \) is indicated by the
* shaped node. The tree in figure 2.7 is pruned in such a way that only those
worlds in which \( A_2 \) is true are accessible to \( q \) in one step from the root of
the tree. The same happens to the nodes that are still accessible to \( q \).

Remember that we were investigating what happened if \( U_{(p,q)}?\text{conditions} \)
were executed. To say something about this we had to see what happened if qleamsA; \( U_{(p,q)}\text{qleamsA}; p\text{chooses} \) was executed. We have seen above what
happens if qleamsA is executed. Now I will look what happens if subsequently
\( U_{(p,q)}\text{qleamsA} \) is executed, that is \( p \) and \( q \) both learn that \( q \) has learned behind
which door the car is.

What happens in this case is a little bit more difficult than what happens
when qleamsA is executed. Let us consider first what happens to \( p \)'s information
state. There are three alternatives \( p \) considers possible. It is unknown to \( p \) in
which of these qleamsA was executed. If it was executed in \( u \), where \( A_1 \) is the
case, \( q \) would have learned that \( A_1 \) is the case and that \( \neg A_2 \) and \( \neg A_3 \) is the
case. If qleamsA was executed in \( v \), where \( A_2 \) is the case, \( q \) would end up in a
possibility like \( \nu_1 \), which is what happened. If it was executed in \( w \), \( q \) would
have learned that \( A_3 \) was the case and that \( \neg A_1 \) and \( \neg A_2 \). Let these possibilities be \( u', v' \) and \( w' \). These are defined as follows.

\[
\begin{align*}
u'(A_1) &= 1 \quad &v'(A_2) &= 1 \quad &w'(A_3) &= 1 \\
u'(A_2) &= u'(A_3) = 0 \quad &v'(A_1) &= v'(A_3) = 0 \\
u'(p) &= v'(p) = w'(p) = \{u', v', w'\} \\
u'(q) &= \{u'\} \quad &v'(q) &= \{v'\} \quad &w'(q) &= \{w'\}
\end{align*}
\]

Now let us consider what happens to \( q \)'s information state. There is only one alternative for \( q \), viz. \( v_1 \). His information about \( A_2 \) does not change, but now he knows \( p \) has learned he has learned the truth about \( A_1, A_2 \) and \( A_3 \). Consequently he knows that the possibilities accessible to \( p \) were updated with this in the way described above. His new information state comprises one possibility, \( v_2 \), which is defined as follows:

\[
\begin{align*}
v_2(A_2) &= 1 \\
v_2(A_1) &= v(A_3) = 0 \\
v_2(p) &= \{u', v', w'\} \\
v_2(q) &= \{v_2\}
\end{align*}
\]

This is the new root of the tree. The whole tree is shown in figure 2.9.
The circled plus symbol indicates \( v_2 \). The pentagon indicates \( u' \), the plus symbol indicates \( u' \) and the triangle indicates \( w' \). When you examine the definitions of \( v_2 \) and \( u' \) closely you will notice that extensionally they are the same. A decoration of the picture in figure 2.9 will assign the same possibility to the circled plus nodes and the plus nodes. We could ‘blow up’ this picture to incorporate the other propositions in \( \mathcal{P} \). If we would make a picture of this it would again be a three dimensional Kripke model. This model is shown in figure 2.10. Remember that in this picture the reflexive and transitive edges are omitted. The dashed lines indicate \( p \)'s accessibility and the normal lines indicate both \( p \)'s and \( q \)'s accessibility. Remember we are still investigating what happens if \( U_{(p,q)\text{conditions}} \) is executed. We have now seen what happens if \( \text{qlearns}A \) and \( U_{(p,q)\text{qlearns}A} \) is executed. Now the only other program that we have to execute is \( \text{pchooses} \).

\[
\text{pchooses} = \\
((\neg C_1; U_{(p,q)}? C_1) \cup (\neg C_1; U_{(p,q)}? \neg C_1)); \\
((\neg C_2; U_{(p,q)}? C_2) \cup (\neg C_2; U_{(p,q)}? \neg C_2)); \\
((\neg C_3; U_{(p,q)}? C_3) \cup (\neg C_3; U_{(p,q)}? \neg C_3))
\]

This program is similar in structure to \( \text{qlearns}A \), but now both \( p \) and \( q \) are updated simultaneously. After this update both \( p \) and \( q \) know which choice has been made. Which pointed Kripke model is the result of this program depends on the pointed Kripke model it is executed in. There are three possible outcomes. These three are shown simultaneously in figure 2.11. Now that \( \text{qlearns}A; U_{(p,q)}\text{qlearns}A; \text{pchooses} \) has been executed we can finally come up with the answer to the question what happens if the program \( U_{(p,q)\text{conditions}} \) is executed. If one views the worlds in the Kripke models shown in figure 2.11 as 27 pointed Kripke models we have to check for example whether \( O_1 \) is true iff...
2.5 An Analysis of the Monty Hall Dilemma

\[ K_q \neg A_1 \land \neg C_1 \land \neg O_2 \land \neg O_3 \] is true in the updated model. We have to see in which worlds the following sentence holds.

\[
\begin{align*}
(O_1 &\leftrightarrow (K_q \neg A_1 \land \neg C_1 \land \neg O_2 \land \neg O_3)) \land \\
(O_2 &\leftrightarrow (K_q \neg A_2 \land \neg C_2 \land \neg O_1 \land \neg O_3)) \land \\
(O_3 &\leftrightarrow (K_q \neg A_3 \land \neg C_3 \land \neg O_1 \land \neg O_2))
\end{align*}
\]

Those worlds where this sentence holds are indicated with filled nodes in figure 2.11. Those worlds where it does not hold are indicated with open nodes. If one would view this cube as a geometric structure, one could view the open nodes as lying on two planes. The first plane contains those points where \( A_i \land O_j \) is true if and only if \( i = j \), that is those points where the opened door is the same as the door the car is behind. The second plane contains those points where \( A_i \land C_j \) is true if and only if \( i = j \), that is those points where the door the player chooses is the same as the door the car is behind. These are the possibilities where \textit{conditions} do not hold. When the update the update \( U_{(p, q)} \) is executed, those possibilities are removed and the remaining possibilities are updated with \textit{conditions}. The resulting Kripke model is shown in figure 2.12. All the transitive and reflexive arrows are omitted again. In fact still all the worlds are connected for both agents, because \( q \text{learns} A_i, U_{(p, q)} \text{learns} A_i \text{ and } p \text{chooses} \textit{consecutively} \) have not actually been executed. The information the agents get when they learn that \textit{conditions} holds is what must be the case after these programs are executed. Note that this Kripke model can simply be obtained from the original Kripke model shown in figure 2.6 by simply removing all those worlds where \textit{conditions} does not hold, if we assume the world where the update was executed in to be one of the remaining worlds. If we would have taken a point which is not one of these worlds, there would have been one more world that was not
accessible to itself but all the other worlds could be reached from it. This is a more general principle concerning group updates that involve the entire group of agents. If an update of the form $U_A?\phi$ occurs, the new Kripke model can be obtained by removing all the arrows that lead to worlds where $\phi$ does not hold. This will be important in chapter 3.

The player and the quiz master have now learned the rules of the quiz and are ready to start the quiz. The only thing that has to happen now is that the quiz master actually learns where the car is and the player and the quiz master both learn that he has learned this, that is $q$learnsA and $U_{(p,q)}q$learnsA are now actually executed. I will not discuss the effect of the execution of these programs in detail, because it is quite similar to what happened to the Kripke model when these programs were ‘hypothetically executed’ when the agents learned that conditions holds. Sufficient to say that the arrows in the direction of the $A_i$ axis will become dashed.

Now the player chooses a door, which means the program pchooses is actually executed. Let us assume for example that the player chose door number one (as in the in the Vos Savant scenario). That means $C_1$ is true and both $C_2$ and $C_3$ are false. Consequently the top layers of the Kripke model are taken off. The resulting Kripke model is shown in figure 2.13. In this picture some more edges are shown. Finally a door is opened. Thus the program qopens is executed, which is very similar to pchooses. It is defined as follows:

$$q\text{opens} = (qO_1; U_{(p,q)}?O_1) \cup (q^-O_1; U_{(p,q)}?^-O_1)) ;$$
$$ (qO_2; U_{(p,q)}?O_2) \cup (q^-O_2; U_{(p,q)}?^-O_2)) ;$$
$$ (qO_3; U_{(p,q)}?O_3) \cup (q^-O_3; U_{(p,q)}?^-O_3))$$

In Marilyn vos Savant’s example this was door number three. The Kripke model is shown in figure 2.14.
Figure 2.13: A Kripke model of the situation after p has chosen a door

Figure 2.14: A Kripke model of the situation after p and q learn which door is opened
2.6 Conclusion

One might well ask what the use of such an analysis is. The last Kripke model might seem somewhat disappointing. Why should we use such a difficult system such as DEL to get such an obvious result? At the end of the quiz it is common knowledge that the player knows the car is behind door number one or door number two, but he does not know which one and the quiz master does know where it is. This is what anyone would say after hearing about the Monty Hall Dilemma. So the analysis does not seem to be very illuminating.

But the point of the analysis is not the final Kripke model. The analysis shows what happens to the information the player and the quiz master have. DEL can handle this quite adequately. Although it may have been difficult to puzzle out what the effect of the execution of a certain program was, the formalization of the rules of the quiz, for instance, was not that difficult. That the result seems intuitively appealing speaks only in favor of the system.

However DEL cannot provide a complete analysis of the Monty Hall Dilemma, for the obvious reason that probability is not incorporated in it. Another disadvantage was already noted, propositional variables cannot change in truth value. One might also wonder whether the programming language is rich enough to capture the intricacies of the Monty Hall Dilemma. In the description of the problem on page 1 the actions are performed by the agents. This cannot be expressed. It is clear that the player chooses a door, but the propositions $C_1$, $C_2$, and $C_3$ have nothing to do with $p$. This could be solved by extending DEL from a propositional logic to a predicate logic, but this is not easy. Even if this could be done, opening and choosing still cannot be seen as programs, which one would like, because they are actions that ‘change the world’ (or that change ‘the information states’) immediately.

Of the problems of DEL mentioned above, I will only try to tackle the problem of probability. In the final analysis we would like to show something like:

\[ \left[ U_{[p,q]}(\text{one car} \land \text{one choice} \land \text{one open}) ; U_{[p,q]}(\text{conditions}; q \text{leaves A}; U_{[p,q]}(\text{q learns A}; \text{p chooses}; \text{opens}) \big( Pr_p(A_1) = \frac{1}{3} \land Pr_p(A_2) = \frac{2}{3} \big) \right] \]

(Assuming the player chooses door number one and door number three is opened.) This is what I will do in the next chapter.
Chapter 3

Probabilistic Dynamic Epistemic Logic

In this chapter I combine probabilistic epistemic logic with dynamic epistemic logic. In section 3.1 I compare the two systems so as to decide which features of the two systems are to be incorporated into the new system. This new system is developed in section 3.2. The concept of adversaries is used in section 3.3 to show how one can add probability to ordinary Kripke models. In section 3.4, two applications of the system are discussed before a new analysis of the Monty Hall Dilemma is given in section 3.5. Three scenarios that are compatible with the description of the Monty Hall Dilemma are discussed, moreover I will show how to reconstruct the mistake one usually makes hearing the Monty Hall Dilemma for the first time. In section 3.6 a sketch is given of a more advanced system for probabilistic dynamic epistemic logic. Finally in section 3.7 some conclusions will be drawn and some open questions will be discussed.

3.1 A Comparison

In this section some of the difficulties facing an attempt to combine the two systems presented in the previous chapters are discussed. On the epistemic level there are three substantial differences between the systems. In the first place the logic for multi-agent systems without probability and DEL differ in the model they use for epistemic logic; Halpern and Tuttle use Kripke models and Gerbrandy uses possibilities. As we saw in chapter 2 this difference is not essential. One can easily change perspective and make a possibility out of a Kripke model and vice versa. The second difference is more fundamental. The Kripke models Halpern and Tuttle use are constructed in the way that was shown in figure 1.2. That means that first of all you have to determine what the local states of the agents are and then you can construct a Kripke model step by step. In DEL you have nothing to do with local states or anything like that. Possibilities are uniquely determined by the truth value they assign to the propositional variables and the information states they assign to the agents. Thirdly in Halpern and Tuttle's system every Kripke model is an $S5$ model, whereas in DEL other models can also be considered. These last two differences indicate that DEL is more general than Halpern and Tuttle's system. We will
have to see which constraints are needed for the combination of these systems.

The two systems also differ in the way they model change. Halpern and Tuttle use the passing of time to model change. As time passes the global state of the system can change. Gerbrandy uses the execution of programs such as updates to model change, however the truth value of propositional variables cannot change. Information change for Halpern and Tuttle means that as time passes the local state of an agent changes. This amounts to a fundamental difference in the way change is modeled in these systems. Halpern and Tuttle are interested in ‘real physical change,’ whereas Gerbrandy is only interested in information change. One could argue that they are interested in two entirely different phenomena. In the analysis of the Monty Hall Dilemma we are more interested in information change.

There is one advantage in Gerbrandy’s way of modeling information change. Once you have constructed a Kripke model for the initial situation, the definition of the programs does the rest. This, in my opinion, is more practical than Halpern and Tuttle’s system, where you have to think far ahead about how the system could develop and what the agents will know as the system develops. The first ingredient of modeling multi-agent systems is determining the local states of the agents. The local state of an agent determines what she knows. Hence, to determine which local state is appropriate for an agent, one has to think about which information an agent has at every point of the development of the system. But the development of a system is defined in terms of changes of the global states, which are determined by local states. Therefore, one has to think long and hard before one makes a decision about the way to model a multi-agent system in Halpern and Tuttle’s system. In DEL you only have to determine which information agents get and translate this into the formal language of DEL. An initial situation can develop in an infinite number of ways, since there are an infinite number of programs and updates that can be executed. In that sense Gerbrandy’s system is more practical and more general too. One could also argue that Gerbrandy’s system is not more general, because in Halpern and Tuttle’s system information changes can occur that cannot be expressed as a program.

The aim of this chapter is to make a combination of these two systems. This combination is to model probability in the way Halpern and Tuttle introduced it and information change in the way Gerbrandy introduced it. That means we would have to use possibility like structures that are pictures of probabilistic Kripke models. The problem with such an approach is that the definition of these structures is circular and the definition of the update operator is circular too. It costs Gerbrandy some trouble to prove that these definitions are correct. A similar construction for probabilistic possibilities is too complex to be incorporated into this chapter, although I will briefly discuss probabilistic non-well-founded structures in section 3.6.

We will have to settle for a simpler system. In chapter 5 of his dissertation Gerbrandy compares DEL to a number of other systems and presents some alternatives to modeling information change. Among them is a system for updates on Kripke models which is sound and complete for a fragment of the full programming language. The idea is that when someone or some group gets information you simply delete the arrows that lead to worlds where that information is not correct. As we saw in chapter 2 this only works if we are dealing with group updates that concern the entire group of agents (see page 46).
Moreover in this limited system the only programs that occur as the ‘subject’ of updates are tests, that is, there are only updates of the form \( U_A ^\phi \). I will try to construct a probabilistic epistemic dynamic logic using this simple system as a starting-point. I will call this new system \( \text{PDEKS} \) (probabilistic dynamic epistemic Kripke semantics).

There is a substantial drawback to \( \text{PDEKS} \). In chapter 2 there were two programs in the analysis of the Monty Hall Dilemma we will not be able to model using this system. In the first place the program \( \text{qlearn} A \) cannot be modeled in \( \text{PDEKS} \), because it contains updates that do not concern the entire group of agents. Secondly the program \( U^{(pa)} \text{qlearn} A \) cannot be modeled, because \( \text{qlearn} A \) is not a test (see page 42). We will have to take a Kripke model where it is common knowledge that \( q \) knows where the car is as the starting point of the analysis. All the other updates mentioned in the analysis of the Monty Hall Dilemma in chapter 2 can be modeled in \( \text{PDEKS} \).

Because we do not take computation trees and the passing of time to model change, the construction of the Kripke models does not depend on local states or anything like that. This gives us the advantage that we are much freer in our choice of Kripke models. Our view is not limited to \( \text{SS} \). \( \text{PDEKS} \) provides a model of probabilistic belief and probabilistic knowledge.

### 3.2 Probabilistic Updates on Kripke Models

To model information change some of the programs Gerbrandy introduced will be used. The definitions will be illustrated by the following example. Suppose there is a vase that contains four red marbles, two green marbles and one black marble. Let us say an agent \( a \) has to pick a marble from this vase. She knows how many marbles the vase contains and how many there are of each colour. What should the probability she assigns to the different outcomes be? This can be modeled using a probabilistic Kripke model \( M = (W, R, V, \mathcal{P}) \); \( R, G \) and \( B \) express that the marble is red, green, or black respectively.

\[
\begin{align*}
W &= \{u, v, w\} \\
R &= \{R_a\} \\
R_a &= W \times W \\
V(R) &= \{u\} \\
V(G) &= \{v\} \\
V(B) &= \{w\} \\
\mathcal{P}(a, u) &= \mathbb{P}(a, v) = \mathbb{P}(a, w) = (S, \mathcal{H}, \mu) \\
S &= W \\
\mathcal{H} &= \{C \mid C \subseteq S\} \\
\mu(\{u\}) &= \frac{4}{7} \\
\mu(\{v\}) &= \frac{2}{7} \\
\mu(\{w\}) &= \frac{1}{7}
\end{align*}
\]

(Because \( \mu \) is a probability measure it is completely defined by the three equations given above.) A picture of this Kripke model is shown in figure 3.1.

Now suppose that \( a \) is colour-blind, that means she cannot distinguish between red and green, but she can distinguish black and not black. Now she picks a marble and she can see that it is not black. It is not difficult to define
a Kripke model for this new situation. The arrows that point to a world where
the marble is black are simply removed and the probability space assignment is
changed accordingly.

\[
\begin{align*}
W &= \{u, v, w\} \\
R &= \{R_a\} \\
R_a &= (W \times W) \setminus \{(x, y) \mid y = w\} \\
V(R) &= \{u\} \\
V(G) &= \{v\} \\
V(B) &= \{w\} \\
\mathcal{P}(a, u) &= \mathcal{P}(a, v) = \mathcal{P}(a, w) = (S', H', \mu') \\
S' &= \{u, v\} \\
H' &= \{C \mid C \subseteq S'\} \\
\mu'\{\{u\}\} &= \frac{4}{9} = \frac{2}{3} \\
\mu'\{\{v\}\} &= \frac{4}{9} = \frac{1}{3}
\end{align*}
\]

A picture of this model is shown in figure 3.2. Later on we shall see why it is
that \(w\) has to be preserved in the set of possible worlds. The question is: can we
define which operation has been performed? If we would analyse this situation
using \textsc{DEL} we would consider this to be the following program:

\[
\text{learn}\mathcal{B} = (\mathcal{B}; U_\alpha \mathcal{B}) \cup (\neg \mathcal{B}; U_\alpha \neg \mathcal{B})
\]

In this case, if we want to say when sentences such as \([\text{learn}\mathcal{B} \mid \text{Pr}_\alpha(G) = \frac{1}{3}]\) are
true, it is obvious that we should extend the language such that programs are
added.

**Definition 3.1 (Language of \textsc{PDEKS}).**

Given a set of agents \(\mathcal{A}\) and a set of propositional variables \(\mathcal{P}\), we define the
sentences and programs of \textsc{PDEKS} simultaneously as follows.

The set of *sentences* of \textsc{PDEKS} is the smallest set that contains \(\mathcal{P}\), and such
that if \(\phi\) and \(\psi\) are sentences, \(\pi\) is a program, \(a\) an agent and \(\alpha \in [0, 1]\), then
\(\neg \phi\), \((\phi \land \psi)\), \(K_a \phi\), \(\text{Pr}_\alpha(\phi) \geq \alpha\) and \([\pi] \phi\) are sentences of \textsc{PDEKS}.
3.2 Probabilistic Updates on Kripke Models

The set of programs is the smallest set that contains $\Box\phi$ and $U\phi$ for each sentence $\phi$ and for which it holds that if $\pi$ and $\pi'$ are programs, then $(\pi;\pi')$ and $(\pi \cup \pi')$ are programs as well.

The abbreviations introduced in definition 1.1 and definition 1.5 are used (pages 5 and 16 respectively). □

The differences between the language of DEL and the language of PDEKS are that sentences of the form $Pr_a(\phi) \geq \alpha$ are added and that updates are limited to tests, moreover the subscript of updates is removed for brevity. After all, only group updates for the entire group will be considered. In DEL updates were executed in possibilities. The question is, what in the present case should be the structure in which an update is executed? As we saw in section 2.3 (page 30) there is a one to one relation between possibilities and bisimulation classes of pointed Kripke models. Therefore the truth definition and the definition of the accessibility relations of programs are all in terms of pointed probabilistic epistemic Kripke models (see definition 1.6, page 17).

**Definition 3.2 (Truth definition for PDEKS)**

Let a model $M = (W, R, V, \mathcal{P})$, a set of propositional variables $\mathcal{P}$ and a set of agents $\mathcal{A}$ be given. For every $p \in \mathcal{P}$, $a \in \mathcal{A}$ and $w \in W$, if $\phi$, $\psi$ are sentences of PDEKS:

\[
(M, w) \models p \quad \text{iff} \quad w \in V(p)
\]

\[
(M, w) \models \phi \land \psi \quad \text{iff} \quad (M, w) \models \phi \text{ and } (M, w) \models \psi
\]

\[
(M, w) \models \neg \phi \quad \text{iff} \quad (M, w) \not\models \phi
\]

\[
(M, w) \models K_a \phi \quad \text{iff} \quad (M, v) \models \phi \text{ if } wR_a v
\]

\[
(M, w) \models Pr_a(\phi) \geq \alpha \quad \text{iff} \quad \mu_{a,w}(S_{a,w}(\phi)) \geq \alpha
\]

\[
(M, w) \models [\pi]\phi \quad \text{iff} \quad (M', w') \models \phi \text{ if } (M, w)[\pi](M', w')
\]

where $S_{a,w}(\phi) = \{v \mid v \in S_{a,w} \land v \models \phi\}$. □

For this definition to work $S_{a,w}(\phi)$ has to be an element of $\mathcal{H}_{a,w}$. otherwise $\mu_{a,w}(S_{a,w}(\phi))$ is not defined. To avoid this problem we could again make an alternative definition using an inner measure (see page 17). The other alternative
is to see which conditions must hold such that every sentence is measurable. For now I will let this matter be. In the analysis of the Monty Hall Dilemma, every sentence is taken to be measurable.

To make the definition complete we have to define the relation \([\sigma]\) for every program \(\sigma\). The only difficulty lies in defining this relation for updates. The other dynamic operators can get their standard interpretation.

\[
(M, w) \models [\sigma] (M', w') \iff (M, w) \models \sigma \text{ and } (M, w) = (M', w')
\]

\[
(M, w) \models [\pi; \pi'(M', w')] \iff \text{there is a } (M'', w'') \text{ such that } (M, w) \models [\pi'] (M'', w'') \text{ and } (M', w') = (M'', w'')
\]

\[
(M, w) \models [\pi \cup \pi'] (M', w') \iff (M, w) \models [\pi](M', w') \text{ or } (M, w) \models [\pi'](M', w')
\]

Neither the accessibility relations nor the probability space assignment change directly as a result of the execution of any of these programs. The only program that can change these is the update. In DEL an update did not change the truth value of the propositional variables. In PDEKS updates should be defined analogously, therefore the set of possible worlds should remain the same and the valuation function should not change either. Only the accessibility relations should change and, of course, the probability space assignment. The accessibility relations should change such that all the arrows that point to a world where the information that is learned is not the case are removed.

How, precisely, should the probability space assignment be updated? In the example with the marbles the sample space and the set of possible worlds accessible to the agent were the same, and exactly the same happened to them. Those worlds where the marble was black were removed from them. Therefore in the general case those worlds where the information that is learned does not hold are removed from the sample space. What happens to the \(\sigma\)-algebra of the probability space when an update is executed? In view of the example given above it would seem that those measurable subsets that contain a world where the marble is black must be removed. One could also say that the operation that was executed on the sample spaces should also be executed on the elements of the \(\sigma\)-algebra.

The most interesting question is how to update the probability measure \(\mu\). It seems we must use conditional probability for this. In the example with the marbles the new probability according to \(a\) for \(a\) to have picked a red marble should equal the earlier conditional probability for it to be red given that it is not black. To cut a long story short, updates in probabilistic epistemic Kripke models can be defined as follows. Different aspects of the definition will be explained on the next page.

**Definition 3.3 (Probabilistic updates)**

Given two pointed Kripke models \((M, w)\) and \((M', v')\) \((M = (W, R, V, \mathcal{P})\) and \(M' = (W', R', V', \mathcal{P}')\)): for all \(a \in A\)

\[
(M, w) \models [U? \phi](M', w') \iff
\]

1. \((M, w) \models [A]\(M', w')\)
2. \(uR_a v\) iff \(uR_v a\) and \((M, v) \models \phi\) and \((M, v) \models Pr_a(\phi) > 0\)
3. \(S_{\text{up}}^{a,v} = \{ s \in S_{a,v} \mid (M, s) \models \phi \text{ and } (M, v) \models Pr_a(\phi) > 0 \} \)
4. \(H_{\text{up}}^{a,v} = \{ H \in H_{a,v} \mid \forall h \in H: (M, h) \models \phi \text{ and } (M, h) \models Pr_a(\phi) > 0 \} \)
3.2 Probabilistic Updates on Kripke Models

5. \( \mu^u_{a,v} (H) = \frac{\mu_a (H)}{\mu_a (S^u_{a,v})} \)

6. \( \Psi' (a, v) = \begin{cases} \exists (a,v) & \text{if } S^u_{a,v} = \emptyset \\ (S^u_{a,v}, \mathcal{H}^u_{a,v}, \mu^u_{a,v}) & \text{otherwise} \end{cases} \)

In this definition \( (M, w)[\mathcal{A}] (M', w') \) is an abbreviation for the phrase \( (M, w) \) and \( (M', w') \) differ at most in the accessibility relations of the members of \( \mathcal{A} \) and the probability space assignment. By the way, conditions 2–6 are independent of the point of the pointed Kripke model.

There are three aspects of this definition that might seem somewhat strange. This is because the definition has to work under the most unlikely circumstances. In the first place it might seem strange that \( W \) and \( W' \) should be the same. Secondly not only arrows are removed that lead to worlds where \( \phi \) is not true, but arrows that lead to worlds where \( Pr_a (\phi) = 0 \) are also removed. In the third place one may wonder why the definition of new probability space assignments is split into two cases. All three of these aspects have to do with borderline cases.

The first and the third aspect both have to do with the possibility that the agent might get false information. If the information an agent got were inconsistent and we would remove every world where that information did not hold, then the set of worlds would be empty which is impossible given the definition of Kripke models. One could also run into trouble modeling the following case: an agent learns that the marble is not black, whereas this information is false and the marble is in fact black. If these worlds where the marble is black are removed, which is the pointed Kripke model the agent is in after the update? Any choice would be unwarranted. Because if \( (M, w)[U ? \phi] (M', w') \), then \( w = w' \). So the point in the new model should be world \( w \), in which the marble would be black. Therefore, to be on the safe side, \( W \) and \( W' \) are the same set.

The second aspect has to do with the borderline case that a probability measure can assign probability 0 to a measurable set. (The measure of \( \mathbb{Q} \) in \( \mathbb{R} \), for instance, equals 0.) There is nothing in the definition of probability spaces that forbids such constructions. The way to deal with such probability spaces is to remove the arrows to those worlds where \( Pr_a (\phi) = 0 \), when \( \phi \) is learned.

The third aspect again concerns learning something that is not true. If the information the agents get is inconsistent every arrow in the model is removed. That means the new sample space should really be empty. But that is impossible given the definition of probability spaces. Therefore the definition is split into two cases. The probability space assignment does not change if the information the agents get is inconsistent. This construction guarantees that there are no gaps in the definition. As was shown in section 1.5 (page 18) the sentence we are really interested in, when it comes to the probability an agent assigns to a sentence \( \phi \), is not \( Pr_a (\phi) \geq \alpha \), but \( K_a (Pr_a (\phi) \geq \alpha) \). As long as there are no worlds accessible where the new probability space assignment is the same as the old, sentences we are interested in are not affected by this construction. An example where this construction proves to be particularly useful for another reason, is given in section 3.4.

A new probability measure \( \mu^u_{a,v} \) assigns to an element \( H \) of \( \mathcal{H}^u_{a,v} \), the old conditional probability of \( H \) given that it is the new sample space. A similar construction was used in chapter 1 to make a probability measure for the agents
in a certain labelled computation tree: the probability that was assigned to an
element \( H \) of the \( \sigma \)-algebra of an agent at a point, equaled the conditional
probability the adversary assigns to \( H \) given that it is in the sample space of
the agent (see section 1.5, page 19). The only problem with this definition
of probabilistic updates is that there is no guarantee the new sample space is
measurable, in that case \( \mu_{\sigma,w}(S_{\sigma,w}^w) \) is not defined. Again we could take the
inner measure (see page 17), which is always defined. It is the same problem
as the problem of the measurability of sentences, because agents can only learn
about the truth value of sentences. If a sentence is measurable (and its measure
is greater than zero), then after an update with that sentence the new sample
space will be measurable.

3.3 Adversaries Revisited

The previous section gave a general theory about the truth value of sentences in
probabilistic epistemic Kripke models and how one should perform an update
in them. I have not yet shown how one can model a specific situation with a
probabilistic Kripke model. This can be done by adding probability to epistemic
logic in a similar fashion as Halpern and Tuttle added probability to multi-
agent systems logic (see section 1.4 and 1.5). The probabilistic epistemic Kripke
models constructed in this way can be seen as a subclass of the Kripke models
of PDEK5. The point of departure is a non-probabilistic Kripke model. We
can use adversaries to generate an a priori probability distribution on the set
of possible worlds. These could in turn be used to define a new Kripke model
without a probability space assignment. Probability space assignments can in
turn be generated by different sample space assignments.

Let us say a Kripke model \( M = (W, R, V) \) is given, where \( W \) is a set of worlds,
\( R \) is a set of accessibility relations, \( V \) is a valuation function, and a set of adver-
saries \( \mathfrak{A} \) is also given. An adversary \( A \) induces a probability space \((S_A, \mathcal{H}_A, \mu_A)\)
on \( W \) such that \( S_A = W \). Given this Kripke model \( M \) and this set of adver-
saries \( \mathfrak{A} \) we want to construct a probabilistic Kripke model \( \mathfrak{M} = (M, \mathfrak{A}, \mathfrak{V}, \mathfrak{Q}) \).
In Halpern and Tuttle’s system the adversary was somehow encoded in the
global states of the system. As we do not have any global states anymore we
have to consider an alternative. Let us say the new set of possible worlds \( \mathfrak{W} \)
consists of pairs of worlds in \( W \) and adversaries in \( \mathfrak{A} \):

\[
\mathfrak{W} = W \times \mathfrak{A}
\]

If one views the set of adversaries to be those that the agents consider possible
the set of accessibility relations can be defined straightforwardly:

\[
(w, A) \mathcal{R}_a(v, B) \iff wR_av
\]

If there is some reason that one agent has more knowledge about the adversaries,
the definition above does not suffice, but I will leave this out of consideration.
It is an interesting question for further research. The valuation function \( \mathfrak{V} \)
only copies the valuation function \( V \).

\[
\mathfrak{V}(p) = \{(w, A) \mid w \in V(p)\}
\]

Now the notorious question which probability space assignments are appropriate
must be put. Again we can construct such an assignment given a sample space
3.3 Adversaries Revisited

A sample space assignment $\mathcal{S}$ assigns a sample space $S_{a,(w,A)}$ to each agent $a$ and pair $(w, A)$. The idea is the same as in Halpern and Tuttle’s system. Given a sample space $S_{a,(w,A)}$ and a subset $S$, the probability assigned to $S$ is the conditional probability a world is in $S$ given that it is in $S_{a,(w,A)}$. At least one requirement has to be satisfied for this idea to work. In Halpern and Tuttle’s system all points in one sample space should be contained in one computation tree, so as to avoid having to do with two or more different adversaries at once ($REQ_1$, page 18). The other requirement was that the sample spaces assigned to the agents at the points should all be measurable and have measures greater than zero ($REQ_2$, page 18). The second requirement implies the first, and because in this new system an analogue of the first requirement is not interesting in itself I will only give an analogue of the second.

**Requirement**

If $\mathcal{S}(a,(w,A)) = S_{a,(w,A)}$, then $S_{a,(w,A)} \in \mathcal{H}_A$ and $\mu_A(S_{a,(w,A)}) > 0$

Consistency can be defined analogously. In Halpern and Tuttle’s system a sample space was consistent if it was a subset of the points the agent considered possible. It is called consistency because in Halpern and Tuttle’s system it implies that $K_a\phi \rightarrow P\phi = 1$ (section 1.5, page 18). We saw in the previous section that in PDEKS a sample space is not a subset of the worlds an agent considers possible if an update is executed with inconsistent information, rendering the set of worlds accessible to an agent empty. So it cannot be guaranteed that the requirement of consistency will hold after an update even if the sample space assignment was consistent before the update. There is however a similar requirement that does hold even if an update with false information is executed.

**Definition 3.4 (Consistency)**

Given a model $\mathfrak{M} = (\mathfrak{M}, \mathcal{A}, \mathcal{B}, \mathfrak{P})$ and a set of agents $\mathcal{A}$, a sample space assignment $\mathcal{S}$ is consistent iff for every $a \in \mathcal{A}$ and every $(w, A) \in \mathfrak{M}$: if $\mathcal{S}(a,(w,A)) = S_{a,(w,A)}$, then

$$S_{a,(w,A)} \subseteq \{(v, B) \mid (w, A) \mathcal{R}_a(v, B)\}$$

or $\{(v, B) \mid (w, A) \mathcal{R}_a(v, B)\} = \emptyset$

It implies that $(K_a\phi \land \neg K_a\neg\phi) \rightarrow (P\phi = 1)$. That means that if the information of an agent $a$ is consistent (in the sense that $\neg K_a \bot$) and $a$ knows that $\phi$ then the probability assigned to $\phi$ is 1. It does not imply that $K_a\phi \rightarrow (P\phi = 1)$. Moreover if a sample space assignment is consistent, then after an update, the new sample space assignment is consistent too.

The set of measurable subsets of a sample space $S_{a,(w,A)}$ comprises those sets that are in the $\sigma$-algebra associated with the adversary such that they are subsets of the sample space $S_{a,(w,A)}$.

$$\mathcal{H}_{(a,(w,A))} = \{S \mid S \in \mathcal{H}_A \text{ and } S \subseteq S_{a,(w,A)}\}$$

The probability measure assigned to a set $S \subseteq S_{(a,(w,A))}$ is the conditional probability with respect to $S_{(a,(w,A))}$:

$$\mu_{(a,(w,A))}(S) = \frac{\mu_A(S)}{\mu_A(S_{(a,(w,A))})}$$
3.4 Ignorance and Learning about Probability

It contains all those worlds with the same adversary as the world the sample space assignment corresponds to. Consequently, every sentence has a probability of either 0 or 1. This is not very interesting. The third sample space assignment in terms of the new system is defined as follows:

\[ \mathcal{E}_{\text{post}}(\omega, A) = \{ \mathcal{E}(\omega, A) \ | \ \omega \in \Omega \} \]

This sample space assignment induces a probability space assignment \( \mathcal{E}_{\text{post}} \) that corresponds with the posterior probability an agent should assign to a sentence. That is the probability an agent assigns to a sentence given all the agent knows, and the probability an agent assigns to a sentence given all the agent's possible worlds, which are in the same computation tree the agent is actually in. In the second sample space assignment, Halpern and Tuttle assigned, \( \mathcal{E}_{\text{post}} \) assigns probabilities other than 0 and 1 to future events. As we have no means of modeling time in \( \mathcal{E}_{\text{post}} \), a real analogue of this sample space assignment cannot be given. The following definition comes closest to it:

\[ \mathcal{E}_{\text{post}}(\omega, A) = \{ \mathcal{E}(\omega, A) \ | \ \omega \in \Omega, A \} \]

We can take the sample space assignments Halpern and Tuttle considered, i.e., \( \mathcal{E}_{\text{prior}} \) and \( \mathcal{E}_{\text{post}} \) (section 1.3) and define analogous sample spaces in the new system.
3.4 Ignorance and Learning about Probability

Which adversaries are appropriate when an agent \( a \) is completely ignorant about a proposition? It is easy to construct a non-probabilistic Kripke model \( M = (W, R, V) \) where \( a \) is ignorant about \( p \).

\[
\begin{align*}
W &= \{w, v\} \\
R &= \{R_a\} = \{W \times W\} \\
V(p) &= \{w\}
\end{align*}
\]

A picture of this Kripke model is shown in figure 3.3. Again the filled node indicates \( p \) is true and the open node indicates \( p \) is false. In this model \( a \) cannot rule out that \( p \) is the case, nor that \( \neg p \) is the case.

If we want to make a probabilistic Kripke model out of this Kripke model, the question is what the appropriate set of adversaries is. One could say that because the agent is ignorant, the probability \( a \) should assign to \( p \) and \( \neg p \) should be the same. Agent \( a \) has no reason whatsoever to think \( p \) is more likely than \( \neg p \). That would mean there is one appropriate adversary, that assigns \( \frac{1}{2} \) to \( p \). If one would generalize this to a rule, it would mean one chooses a uniform probability distribution in the face of ignorance.

One could also argue that this does not provide an adequate model of the situation where \( a \) is completely ignorant about \( p \), because the agent must also be ignorant about the probability of \( p \) and \( \neg p \). The agent cannot rule out any adversary. If somehow the agent would learn that the probability of \( p \) is in fact \( \frac{1}{2} \), this is not inconsistent with the information the agent had. But, if we would apply the rule as it was suggested above, it would be. Therefore the appropriate set of adversaries must contain every conceivable adversary (such that all sentences are measurable). This yields a set of adversaries \( \mathcal{A} \) that is not countable, because the probability of \( p \) can be anywhere in the interval \([0, 1]\).

Thus the set of worlds of the probabilistic epistemic Kripke model is not countable. All these worlds are accessible to \( a \), and if we would take \( S^{post} \) as the probability space assignment, each sample space would contain exactly two points, the probability assigned to these points is the same probability the adversaries assign to them. Note that the definition of probabilistic updates guarantees that if \( a \) would learn \( p \) the arrows that point to the two worlds induced by the adversary that assigns probability 0 to \( p \) are removed.

This issue seems to pertain to non-monotonic reasoning. The standard example of this is the following. Suppose you acquire the information that Tweety is a bird. You will presume that Tweety can fly. Then you learn that Tweety is an emu, a cassowary, a kiwi, a rhea, a penguin or an ostrich. You draw the conclusion that your presumption is wrong, and that Tweety cannot fly. The same happens when someone gets information and has to presume that there is a probability distribution. If someone tells you that there is an island inhabited by knights and knaves, you presume that, if you are on the island the chances of
running into a knight is equal to the chance of running into a knave. However, this is only a presumption, you would not say that someone is lying if she goes on to tell you that for each knight there are ten knaves. I think non-monotonicity also plays an important role in the Monty Hall Dilemma. However, I will not try to devise a non-monotonic probabilistic logic. All the information the player and the quiz master have about probabilities must have been learned explicitly.

The second problem concerns learning about probability. What happens if an update like \( U?(Pr_\alpha \phi \geq \alpha) \) is executed. Let us consider the Kripke model which is shown in figure 3.3. Let us say that \( \alpha \) considers two adversaries possible: one that assigns \( \frac{1}{2} \) to \( p \) and one that assigns \( \frac{1}{3} \) to \( p \). With these probabilities a probabilistic epistemic Kripke model can be constructed. This model is shown in figure 3.4. Formally this Kripke model is defined as follows. \( \mathfrak{M} = (\mathcal{W}, \mathcal{R}, \mathfrak{V}, \mathfrak{P}) \)

\[
\mathcal{W} = \{(w, A), (v, A), (w, B), (v, B)\}
\]
\[
\mathcal{R} = \{\mathcal{R}_a\} = \{\mathcal{W} \times \mathcal{W}\}
\]
\[
\mathfrak{V}(p) = \{(w, A), (w, B)\}
\]
\[
\mathfrak{S}_A = \mathfrak{S}_B = \{w, v\}
\]
\[
\mathcal{H}_A = \mathcal{H}_B = \{\emptyset, \{w\}, \{v\}, \{w, v\}\}
\]
\[
\mu_A\{w\} = \frac{1}{2}, \quad \mu_A\{v\} = \frac{1}{7}, \quad \mu_B\{w\} = \frac{1}{3}, \quad \mu_B\{v\} = \frac{7}{9}
\]
\[
\mathfrak{S}(a, (w, A)) = \mathfrak{S}(a, (v, A)) = \{(w, A), (v, A)\}
\]
\[
\mathfrak{S}(a, (w, B)) = \mathfrak{S}(a, (v, B)) = \{(w, B), (v, B)\}
\]

\( \mathfrak{P} \) is the probability space assignment induced by \( \mathfrak{S} \) (which is \( \mathfrak{S}^{\text{Post}} \)). What happens if the program \( U?(Pr_\alpha \phi = \frac{1}{5}) \) is executed in the pointed Kripke
Figure 3.5: The Kripke model after the update $U?Pr_a(p) = \frac{1}{3}$

model $(\mathcal{M}, (w, B))$? This yields a new pointed Kripke model $(\mathcal{M}', (w, B))$. To construct it we simply have to follow the definitions from section 3.2. Consequently the only things that can change are the accessibility relations and the probability space assignment. The new Kripke model is shown in figure 3.5. In the new picture it can easily be seen that all the arrows pointing to a possible world where $Pr_a(p) = \frac{1}{3}$ does not hold are removed. The probability space assignment however stays exactly the same. This is because in every world in $\mathcal{G}(a, (w, B))$ and $\mathcal{G}(a, (v, B))$ the sentence $Pr_a(p) = \frac{1}{3}$ is true. Therefore the updated probability space is the same as the old one. However in every world in $\mathcal{G}(a, (w, A))$ and $\mathcal{G}(a, (v, A))$, $Pr_a(p) = \frac{1}{3}$ is not true. Consequently the new sample space would be empty. But the gap filling construction in the definition of probabilistic updates prevents this, therefore in this case the new probability space is equal to the old one. The result is exactly what we want.

$$(\mathcal{M}, (w, B)) \models \neg K_aPr_a(p) = \frac{1}{3}$$

$$(\mathcal{M}, (w, B)) \models \neg K_a\neg Pr_a(p) = \frac{1}{3}$$

$$(\mathcal{M}', (w, B)) \models K_aPr_a(p) = \frac{1}{3}$$

In the old model the agent did not know whether the probability she should assign to $p$ was $\frac{1}{3}$ or not. After the update she knows it is $\frac{1}{3}$. Consequently:

$$(\mathcal{M}, (w, B)) \models [U?Pr_a(p) = \frac{1}{3}]K_aPr_a(p) = \frac{1}{3}$$

The update is successful. After $a$ learns the probability she should assign to $p$ is $\frac{1}{3}$, she knows she should assign $\frac{1}{3}$ to it. Note that if we had taken any other pointed Kripke model based on $\mathcal{M}$ the result would have been similar. Thus PDKS gives an adequate model of information change, when the information is propositional, involves knowledge or when the information concerns probability.
3.5 An Analysis of the Monty Hall Dilemma

Now we can make an analysis of the Monty Hall Dilemma using PDEKS. We can take the same set of agents and propositional variables as in chapter 1 and chapter 2. Thus $\mathcal{A} = \{p, q\}$ and $\mathcal{P} = \{A_1, A_2, A_3, C_1, C_2, C_3, O_1, O_2, O_3\}$. Again the idea of the analysis is to model what happens to the information the player and the quiz master have. It is interesting to note that in the actual description of the Monty Hall Dilemma (on page 1) probability is not mentioned. No probability distribution of the car over the doors is mentioned. Moreover the question that is actually asked is whether it is advantageous to switch doors, but not what the probability is that the car is behind door number two, for instance.

Marilyn vos Savant’s answer is probabilistic. It is not surprising that we have to make some assumptions about probability to get the answer Vos Savant gave. In section 3.5.1 I shall display these assumptions. In section 3.5.2 I shall expose the consequences of what happens if we do not take these assumptions for granted. The mistake one is inclined to make when hearing the Monty Hall Dilemma for the first time is discussed in section 3.5.3.

The starting point for all scenarios is a normal epistemic Kripke model. With an appropriate set of adversaries a probabilistic epistemic Kripke model can be constructed. In the construction of the epistemic Kripke model I will have to make one assumption. As was said in the beginning of this chapter we cannot model the effect of the programs $\text{qlearns}A$ and $U_{\{p,q\}}\text{qlearns}A$. The effect of these programs being executed consecutively was that it became common knowledge that the quiz master knew the truth about $A_1$, $A_2$ and $A_3$.

That is:
$$
C_{\{p,q\}} ((A_1 \rightarrow K_q A_1) \land \\
(A_2 \rightarrow K_q A_2) \land \\
(A_3 \rightarrow K_q A_3) \land \\
(\neg A_1 \rightarrow K_q \neg A_1) \land \\
(\neg A_2 \rightarrow K_q \neg A_2) \land \\
(\neg A_3 \rightarrow K_q \neg A_3))
$$

Because we cannot model these programs we will simply have to assume that this sentence holds as a premise. Further I will assume nothing. This means that both $p$ and $q$ know nothing about the truth value of $C_1$, $C_2$, $C_3$, $O_1$, $O_2$, and $O_4$. Moreover $p$ knows nothing about the truth value of $A_1$, $A_2$, and $A_3$. So for $p$ there are $512$ $(2^9)$ worlds accessible. But for $q$ in each world there are only $64$ $(2^6)$ worlds accessible. Again I will assume the model is an $55$ model.

What is the appropriate set of adversaries we should consider? I will assume that $p$ and $q$ are completely ignorant, so they consider every adversary possible (such that every sentence is measurable). So now we have a model with a non-countable number of what from now on will be called simply ‘worlds’ (namely, pairs consisting a world from the original model and an adversary.)

3.5.1 The standard scenario

What are the rules of the quiz in the standard scenario? The players still learn that there is only one car behind the doors, only one choice can be made, and

\(^1\)Although common knowledge was not formally introduced in the language, this can easily be done. Because we are dealing with Kripke models common knowledge gets its standard interpretation (see section 1.1).
only one door can be opened. That is the update $U? (\text{onecar} \land \text{onechoice} \land \text{oneopen})$ is executed. As a reminder:

\[
\begin{align*}
\text{onecar} &= \bigvee_{1 \leq i \leq 3} (A_i \land \bigwedge_{j \neq i, 1 \leq j \leq 3} \neg A_j) \\
\text{onechoice} &= \bigvee_{1 \leq i \leq 3} (C_i \land \bigwedge_{j \neq i, 1 \leq j \leq 3} \neg C_j) \\
\text{oneopen} &= \bigvee_{1 \leq i \leq 3} (O_i \land \bigwedge_{j \neq i, 1 \leq j \leq 3} \neg O_j)
\end{align*}
\]

If we make a picture of one of the sample spaces in the Kripke model, we get the well known Kripke cube, shown in figure 3.6. Remember that there are non-countably many of these sample spaces that are all accessible to $p$ and $q$. The normal arrows indicate accessibility for both $p$ and $q$. The dashed arrows indicate accessibility for $p$ only. The arrows pointing to and from other sample spaces have all been omitted. Again this picture should really be regarded as a pointed Kripke model. I have not indicated what probability $p$ and $q$ assign to each of these worlds because they do not have any information about this yet. The question is whether the player and the quiz master learn anything about the probability distribution? I think that a non-monotonic reasoner would assume that the probability the car is behind a given door is $\frac{1}{3}$ if she only had the information that the car is behind one of the three doors. That means the player and the quiz master both learn the following:

\[
equaldoors = (Pr_p(A_1) = \frac{1}{3}) \land (Pr_p(A_2) = \frac{1}{3}) \land (Pr_p(A_3) = \frac{1}{3})
\]
Figure 3.7: One sample space of the Kripke model of the situation after $U?\text{conditions}^-$ has been executed

The quiz master already knows where the car is. Hence the probability $p$ should assign to these events is used. The effect of the program $U?\text{equaldoors}$ being executed is that arrows pointing to worlds where $\text{equaldoors}$ does not hold, are removed, just as in the second example discussed in section 3.4.

Now $p$ and $q$ have to learn under what conditions the quiz master may open a door. Because the programs $\text{qlearnsA}$ and $U_{[p,q]}\text{qlearnsA}$ cannot be modeled we have to define these anew. To distinguish these from the original conditions I will call these conditions$^-$.

$$
\text{conditions}^- = \{p \text{chooses} \}
((O_1 \leftrightarrow (K_q \neg A_1 \land -C_1 \land -O_2 \land -O_3))\land
(O_2 \leftrightarrow (K_q \neg A_2 \land -C_2 \land -O_1 \land -O_3))\land
(O_3 \leftrightarrow (K_q \neg A_3 \land -C_3 \land -O_1 \land -O_2))
$$

The effect of the execution of the program $U?\text{conditions}^-$ is quite similar to the effect of the execution of the program $U_{[p,q]}?\text{conditions}$. Therefore I will not discuss it here. A picture of one resulting sample space is shown in figure 3.7. Note again that this is a picture of a sample space. The worlds where conditions$^-$ did not hold are removed from the sample spaces, although they are not removed from the set of possible worlds.

There are some aspects of the puzzle that remain implicit in the informal description of the Monty Hall Dilemma. One important aspect is that the choice that the player makes is independent of where the car is. A classical way to express independence would be something like $Pr_p(A_1 \land C_j) = Pr_p(A_1)Pr_p(C_j)$; the probability $p$ should assign to $A_1 \land C_j$ is the product of the probability he should assign to $A_1$ and the probability he should assign to $C_j$. However such sentences are not in the language of $\text{PDEKS}$. $Pr_p(A_1)$ is not interpreted as a term, nor can identity between terms or multiplication be expressed. There is
another way to express independence.

\[ \bigwedge_{1 \leq i \leq 3} (Pr_p(A_i) = \alpha) \iff [\text{chooses}] (Pr_p(A_i) = \alpha) \]

The probability \( p \) assigns to the car being behind a particular door after he has chosen a door is the same as the probability he assigned to it before. In this sentence however \( \alpha \) must be read as a variable ranging over \([0, 1]\). Then \( Pr_p(A_i) \) should really be regarded as a term again. The language, as it presently stands, is simply not rich enough to express independence in general. However if there are a finite number of adversaries or if the model contains only a finite number of sample spaces, then such a sentence can simply be seen as a conjunction. Hence in such a case independence can be expressed. In this case however, we had already assumed that the probability \( p \) assigned to the car being behind one particular door is \( \frac{1}{3} \); therefore we can simply execute an update with independentAC:

\[ \text{independentAC} = \bigwedge_{1 \leq i \leq 3} [\text{chooses}] (Pr_p(A_i) = \frac{1}{3}) \]

The player does not learn anything about the location of the car by choosing a door. There is only one more thing that has to be assumed; when the quiz master can choose between opening two doors (when the player has picked the door with the car behind it), then he will not have any preferences regarding the two doors. The probability he opens one is equal to the probability he opens the other.

\[ \text{nopreference} = \bigwedge_{1 \leq i < j \leq 3} [\text{chooses}](\neg K_q \neg O_i \land \neg K_q \neg O_j \rightarrow (Pr_q(O_i) = \frac{1}{2} \land Pr_q(O_j) = \frac{1}{2})) \]

The effect of the update \( U?((\text{independentAC} \land \text{nopreference}) \land \neg K_q \neg O_i \land \neg K_q \neg O_j \rightarrow (Pr_q(O_i) = \frac{1}{2} \land Pr_q(O_j) = \frac{1}{2})) \) is shown in figure 3.8. The effect of learning that the probability the player assigns to the car being behind a particular door is the same after he has chosen a door, is that those arrows that point to worlds where that is not the case are removed. Consequently if one would take a ‘C1 layer’ from a sample space, one would see that the probability assigned to the set of worlds where \( A_1 \) holds is equal to the probability assigned to the set of worlds where \( A_2 \) holds, and to the probability assigned to the set of worlds where \( A_3 \) holds. Because in each layer the probability assigned to the car being behind a particular door is equal, the conditional probability the car is behind a particular door given a choice of the player is \( \frac{1}{3} \).

The effect of learning that the quiz master has no preference regarding the door he opens, is that if one would take a ‘C1 layer’ and see that, for a given \( A_j \) there are two distinct worlds, the probability assigned to one of these worlds would be equal to the probability assigned to the other. Consequently if the door the player chooses is the same door that contains the car, the probability the quiz master will open one of the remaining doors is the same as the probability he will open the other remaining door.

As an example I have chosen the sample space assigned to the player where the probabilities of \( C_1 \) are also \( \frac{1}{3} \), this is an arbitrary choice. We do not have to assume a probability distribution over the choices of the player. (The player
can have a lucky number, for instance.) The agents have not acquired any information about the prior probability of the choice of the player. Note that the probabilities attached to the nodes are the probabilities the player should assign to them, not the quiz master! Now the agents have learned the rules of the quiz, and so the quiz can begin.

Now that they are on stage the player chooses a door, that is the program *pchooses* is executed. Let us assume again that it is the first door, just as in vos Savant’s version of the Monty Hall Dilemma. Consequently the top layers of the sample spaces are removed. The resulting sample space is shown in figure 3.9. The probability space assignment is adjusted accordingly. Note again that only the probability the player assigns to these worlds is shown. Note also that this picture can be seen as any sample space that contains worlds that

![Diagram](image-url)

Figure 3.8: One sample space of the Kripke model of the situation after $U?(\text{independentAC} \land \text{nopreference})$ has been executed

![Diagram](image-url)

Figure 3.9: One sample space of the Kripke model of the situation after *pchooses* has been executed
are still accessible to the agents. Because if a sample space is still accessible, the conditional probability assigned by the adversary to the remaining worlds given all the information the agents have acquired must be equal the probability assigned to them in the sample space shown in figure 3.9.

Finally the quiz master opens a door; the program qopens is executed. Let us assume that it is door number three. The final sample space is shown in figure 3.10. The resulting probabilities that are assigned to the worlds are $\frac{1}{3}$ and $\frac{2}{3}$. Consequently the sentence $K_p(Pr_p(A_1) = \frac{1}{3})$ holds and the sentence $K_p(Pr_p(A_2) = \frac{2}{3})$ holds. This is the result we were looking for. We have shown that:

$$
C_{(p,q)}((A_1 \rightarrow K_q A_1) \land (A_2 \rightarrow K_q A_2) \land (A_3 \rightarrow K_q A_3) \land
\neg A_1 \rightarrow K_q \neg A_1) \land \neg A_2 \rightarrow K_q \neg A_2) \land (\neg A_3 \rightarrow K_q \neg A_3) =
[U_{(p,q)};\text{onecar} \land \text{onechoice} \land \text{oneopen}] ;
U_{(p,q)};\text{equaldoors}; U_{(p,q)};\text{conditions} ;
U_{(p,q)};\text{independentAC} \land \text{nopreferences}] ;
pchooses; qopens;((Pr_p(A_1) = \frac{1}{3}) \land (Pr_p(A_2) = \frac{2}{3}))
$$

### 3.5.2 Laziness

In this section I will look at some other scenarios, most of which have to do with laziness. These scenarios are not disqualified by the informal description of the Monty Hall Dilemma, but most readers of the puzzle will not think of them. The first has to do with lazy stagehands, the second with lazy quiz masters.

The lazy stagehands scenario shows what surprising results may be reached when one does not assume that the probability the car is behind a particular door is $\frac{1}{3}$. What if there are two stagehands, Larry and Prudence. Larry is very lazy and prudence is very conscientious. Let us say that the quiz is broadcast six days of the week. Being a stagehand is a part time job. Larry works three days of the week and Prudence works the other three days. Being lazy, Larry always places the car behind the first door. This door is closest to the entrance of the studio where the cars are brought in. He lures the goats, that roam free in the studio, to door number two and three by scattering some food behind
those doors.

Prudence proceeds in quite a different fashion. She has a bag containing three playing cards: a jack, a queen and a king. When she wants to decide where to put the car she draws one of the cards from her bag. If it is a jack, she puts it behind door number one; if it is a queen she puts it behind door number two and if it is a king she puts it behind door number three. She places the goats behind the other doors.

Let us also assume that which days of the week they work is completely random. The probability the car is behind door number one must be $\frac{2}{3}$ ($\frac{4}{3} + \frac{1}{3}$). The probability the car is behind door number two must be $\frac{1}{3}$, the same holds for door number three. If we would give this information to the player and the quiz master before the quiz started the result is quite surprising. Ceteris paribus, if door one is chosen and door three is opened, the end result would be that the probability the player should assign to the car being behind door number one still equals $\frac{2}{3}$, and he should assign $\frac{1}{3}$ to the other door. This is exactly the opposite result compared to the result in the standard scenario. The player should always pick door number one, and never switch doors.

Another scenario that can be thought of introduces a lazy quiz master. The lazy quiz master does not like to walk. If it were up to him an assistant would open the doors for him, but the producers are a bit cheap, and they make him open the doors himself. Let us assume that door number one is nearest to him and that door number three is farthest away. If he has to open a door, he opens the door nearest to him (such that the rules of the quiz allow him to open it). Obviously he does not like Larry a lot, because when Larry is working he can never open door number one. But let us assume for now that Prudence is working full-time. Now if the player chooses door number one and the lazy quiz master walks all the way to door number three to open it, the player must surely switch, because the player can be sure the car is behind door number two (Assuming he knows the quiz master is lazy).

On the other hand we could also think of a scenario where the quiz master is an athlete. He grabs every possible chance of getting a little exercise. If he has to open a door he will run to the door that is farthest away. He loves to work when Larry is working. But let us assume again that Prudence is working full-time. Now if the player chooses door number one and the quiz master opens door number three, the chances the car is behind door number one are equal to the car being behind door number two. However, whatever the quiz master is lazy or athletic, if Prudence is working, you will never decrease your chances of winning by switching doors.

### 3.5.3 Where the experts go wrong

Those people like Von Saher (page 2), who think the chances of winning do not increase by switching doors, do not argue that the quiz master or the stagehand is lazy. Nor do they take another sample space assignment. I suspect that they do not regard opening a door learning about a proposition like $O_3$, but they regard it as learning the truth about door number 3. That is, opening door number three means the following program is executed.

\[
\text{sesame3} = (A_3; U?A_3) \cup (\neg A_3; U?\neg A_3)
\]
3.5 An Analysis of the Monty Hall Dilemma

This gives a good formalisation of what happens if door number three is opened. Hence you do not need the propositions $O_1$, $O_2$, and $O_3$. The undesired consequence is that the quiz master nor the player can learn the conditions about opening the doors. If one would analyze the Monty Hall Dilemma like this the set of propositional variables $\mathcal{P} = \{A_1, A_2, A_3, C_1, C_2, C_3\}$. The programs that are executed would be $U_{(p,q)}(\text{onecar} \land \text{onechoice})$, $U_{(p,q)}(\text{equaldoors})$, $U_{(p,q)}(\text{independentAC})$, pchooses and sesame3. Before the choice is made and door three is opened a sample space would look like the sample space in figure 3.11. Again it is assumed that it is common knowledge that the quiz master knows where the car is, hence the dashed lines. This is a sample space where the chance that the player chooses a particular door is $\frac{1}{3}$. Now if the player chooses a door, that is if pchooses is executed, the tops of the sample spaces are removed, leaving only the lowest line. This is shown in figure 3.12. This can be seen as any of the remaining sample spaces, because in each of them the probability assigned to a particular door should be $\frac{1}{3}$. This is the same probability distribution over the doors after the player has chosen a door as in the analysis presented in section 3.5.1, the only difference is that there were two worlds where $A_1$ was the case, because two different doors could be opened. If sesame3 is executed, and there is a goat behind door number three, the ratio between the chances for $A_1$ and $A_2$ remains the same. So the chance that the car is behind door number one is $\frac{1}{3}$, as is shown in figure 3.13. This is exactly the result that most people arrive at, when they hear the Monty Hall Dilemma for the first time. To me it seems that this is not a matter of foolishness, but a mistake anyone is liable to make. That a door will be opened is the last piece of
information one gets. It is difficult to see what probabilities should be assigned to opening the various doors, if you have not taken into account from the very beginning that a door will be opened. For simplicity one tends to take the simplest approach; one assumes that which door is opened is completely random. From the analysis given in section 3.5.1 it is clear that this is not the case. The analysis in this section would be correct if it were completely random. If for example lazy Larry has build the stage, but he has not done a very good job such that door number three simply collapses, thus revealing its contents, this analysis is correct.

3.6 Likelihoods

The analysis of the Monty Hall Dilemma with PDEKS was quite satisfactory. A substantial drawback of the analysis was that the programs qlearnSA and $U_{(P, Q)}$qlearnSA are not in the language. If one could also give a probabilistic account of these sorts of updates, one could truly say that a general system for reasoning about knowledge, probability and information change has been given. As a start we would have to define non-well-founded structures such that the decoration of a probabilistic epistemic Kripke model is such a structure. This is completely analogous to the definition of possibilities with respect to normal Kripke models. I will call these structures likelihoods.

**Definition 3.5 (Likelihoods)**

Let $\mathcal{A}$, a set of agents, and $\mathcal{P}$, a set of propositional variables, be given. The class of **likelihoods** is the largest class such that:

- A likelihood $l$ is a function that assigns to each propositional variable $p \in \mathcal{P}$ a truth value $l(p) \in \{0, 1\}$ and to each agent $a \in \mathcal{A}$ an information state $l(a)$.

- An information state is a pair $(L, R)$ such that $L$ is a set of likelihoods and $R$ is a probability space such that its sample space is a set of likelihoods.

I will use $L_{a,t}$ to indicate the set of likelihoods that are accessible to $a$ in $l$. That is if $l(a) = (L, R)$, then $L = L_{a,t}$. Similarly $R_{a,t}$ is $a$’s probability space in $l$; if $l(a) = (L, R)$, then $R_{a,t} = R$. Analogously $R_{a,t} = (S_{a,t}, \mathcal{H}_{a,t}, \mu_{a,t})$.

I call the probabilistic dynamic epistemic logic that can be made with these structures PDEL. A decoration function $\delta$ of a probabilistic epistemic Kripke model can be defined as follows. Let a set of propositional variables $\mathcal{P}$, a set of agents $\mathcal{A}$ and a probabilistic epistemic Kripke model $(W, R, V, \mathcal{P})$ be given. For every $c \in W$, $p \in \mathcal{P}$ and $a \in \mathcal{A}$:
3.6 Likelihoods

\[
\begin{align*}
\delta(c)(\mu) &= 1 \text{ iff } c \in V(p) \\
\delta(c)(a) &= (L, (S, \mathcal{H}, \mu)) \text{ iff } \\
L &= \{\delta(m) \mid cR_a m\} \\
S &= \{\delta(m) \mid m \in S_{a,c}\} \\
\mathcal{H} &= \{H' \mid H \in \mathcal{H}_{a,c} \wedge H' = \{\delta(m) \mid m \in H\}\} \\
\mu(H') &= \max \{\mu_{a,c}(H) \mid H \in \mathcal{H}_{a,c} \text{ and } H' = \{\delta(m) \mid m \in H\}\}
\end{align*}
\]

Why the probability measure \(\mu\) is constructed in this way will be explained later.

The definition of likelihoods can be illustrated by the following example. How should one model the situation where an agent \(a\) does not know whether \(p\) is the case, but she does know the probability that \(p\) is the case is \(\frac{1}{2}\)? This can be modeled using two likelihoods \(l\) and \(m\):

\[
\begin{align*}
l(p) &= 1 \\
l(a) &= (\{l, m\}, \bar{R}) \\
m(p) &= 0 \\
m(a) &= (\{l, m\}, \bar{R}) \\
\bar{R} &= (S, \mathcal{H}, \mu) \\
S &= \{l, m\} \\
\mathcal{H} &= \{\emptyset, \{l\}, \{m\}, \{l, m\}\} \\
\mu(\emptyset) &= 0 \\
\mu(\{l\}) &= \frac{1}{2} \\
\mu(\{m\}) &= \frac{1}{2} \\
\mu(\{l, m\}) &= 1
\end{align*}
\]

A picture of these likelihoods is shown in figure 3.14. The filled node indicates \(l\), the open node indicates \(m\). Although this is a graphical representation of likelihoods, likelihoods cannot be represented graphically in an elegant way in general. The same goes for probabilistic Kripke models.

To avoid the following problem the probability measure is constructed as it is. What if there is a Kripke model where there are two distinct possible worlds that have the same valuation function and the same worlds are accessible to the agents? A picture of such a Kripke model is given in figure 3.15. Note that in this Kripke model the information the agent has is the same as in figure 3.14. If such a Kripke model were decorated such that the decoration functions are possibilities, they get the same decoration. So, as possibilities these worlds are the same. Similar things happen with likelihoods. The nodes \(m_1\) and \(m_2\) picture

Figure 3.14: A picture of two likelihoods
the same likelihood. However, probability is assigned to them separately. The question is, if they are the same likelihood, what probability should be assigned to this likelihood. It seems the probability that was assigned to the set that contained exactly those two possible worlds should be assigned to the likelihood. To achieve this, \( \mu(H') \) has to be defined as the maximum of the probabilities that are assigned to sets \( H \) that comprise those possible worlds that are decorated in such a way that they yield \( H' \). In this case the probability assigned to likelihood \( m \) (i.e. the decoration of \( m_1 \) and \( m_2 \)) should equal \( \frac{1}{2} \).

We can easily make a truth definition for the language for probabilistic epistemic logic (definition 1.5) in terms of likelihoods.

**Definition 3.6 (Static truth definition for PDEL)**

Let a set of propositional variables \( \mathcal{P} \), a set of agents \( \mathcal{A} \) and a likelihood \( l = (L, \mathcal{R}) \) be given. For every \( p \in \mathcal{P}, \alpha \in \mathcal{A} \) and \( \phi, \psi \in \mathcal{L}^x_{\mathcal{P}, \mathcal{A}} \):

\[
\begin{align*}
    l \models p & \quad \text{iff} \quad l(p) = 1 \\
    l \models \phi \land \psi & \quad \text{iff} \quad l \models \phi \text{ and } l \models \psi \\
    l \models \neg \phi & \quad \text{iff} \quad l \not\models \phi \\
    l \models K_\alpha \phi & \quad \text{iff} \quad \forall m \in L : m \models \phi \\
    l \models Pr_\alpha(\phi) & \quad \text{iff} \quad \mu_\alpha, l(S_{\alpha,l}(\phi)) \leq \alpha
\end{align*}
\]

where \( S_{\alpha,l}(\phi) = \{ m \mid m \in S_{\alpha,l} \land m \models \phi \} \). \( \square \)

For this definition to work, \( S_{\alpha,l}(\phi) \) has to be an element of \( \mathcal{H}_{\alpha,l} \); otherwise \( \mu_\alpha, l(S_{\alpha,l}(\phi)) \) is not defined. To avoid this problem we could again make an alternative definition using an inner measure (see page 17) or we would have to see under which conditions every sentence is measurable. Once more I will leave this question aside.

To model information change in likelihoods I will simply use the programs Gerbrandy introduced. The definitions will be illustrated by the example that was used in section 3.2. Suppose there is a vase that contains four red marbles, two green marbles and one black marble. Let’s say an agent \( \alpha \) picks a marble from this vase. She knows how many marbles the vase contains and how
3.6 Likelihoods

Figure 3.16: Three likelihoods

many there are of each colour. Let us suppose again that she is colour blind. What should the probability she assigns to the different events be? This can be modeled using three likelihoods \( k, l, m \). Let \( \mathcal{P} = \{R, G, B\} \) (expressing that the marble is red green or black respectively) and \( \mathcal{A} = \{a\} \).

\[
\begin{align*}
  k(R) &= 1 \\
  k(G) &= k(B) = 0 \\
  l(G) &= 1 \\
  l(R) &= l(B) = 0 \\
  m(B) &= 1 \\
  m(R) &= m(G) = 0 \\
  L_{a,k} &= L_{a,l} = L_{a,m} = \{k, l, m\} \\
  \mathcal{H} &= \{S, \mathcal{H}, \mu\} \\
  S &= \{k, l, m\} \\
  \mathcal{H} &= \{C \mid C \subseteq S\} \\
  \mu(\{k\}) &= \frac{1}{7} \\
  \mu(\{l\}) &= \frac{2}{7} \\
  \mu(\{m\}) &= \frac{1}{7}
\end{align*}
\]

(Because \( \mu \) is a probability measure it is completely defined by the three equations given above.) A picture of these likelihoods is shown in figure 3.16. The dot-shaped node will be decorated with \( k \), the square shaped node will be decorated with \( l \) and the x shaped node will be decorated with \( m \). Now she picks a marble and she can see that it is not black, let us assume that this is true. It
Figure 3.17: A picture of the situation after $a$ has learned the marble is not black.

is not difficult to define the likelihoods that model this new situation.

$$
\begin{align*}
    k'(R) & = 1 \\
    k'(G) & = k'(B) = 0 \\
    l'(G) & = 1 \\
    l'(R) & = l'(B) = 0 \\
    L_{a,\nu} & = L_{a,\nu'} = \{k', l'\} \\
    \mathcal{R}_{a,\nu} & = \mathcal{R}_{a,\nu'} = (S', \mathcal{H}', \mu') \\
    S' & = \{k', l'\} \\
    \mathcal{H}' & = \{B \mid B \subseteq S'\} \\
    \mu'(\{k'\}) & = \frac{4}{9} = \frac{2}{3} \\
    \mu'(\{l'\}) & = \frac{2}{9} = \frac{1}{3}
\end{align*}
$$

A picture of these is shown in figure 3.17. The question is: can we define which operation is performed? If we would analyse this situation using DEL we would consider this to be the following update:

$$
\text{learnB} = U_a((?B; U_a ?B) \cup (?\neg B; U_a ?B))
$$

Now we extend the language to contain the entire programming language.

**Definition 3.7 (Language of PDEL)**

Given a set of agents $A$ and a set of propositional variables $P$, we define the sentences and programs of PDEL simultaneously as follows.

The set of **sentences** of PDEL is the smallest set that contains $P$, and such that if $\phi$ and $\psi$ are sentences, $\pi$ is a program, $a$ an agent and $\alpha \in \{0, 1\}$, then $\neg\phi$, $(\phi \land \psi)$, $K_a\phi$, $(Pr_a(\phi) \geq \alpha)$ and $[\pi]\phi$ are sentences of PDEL.

The set of **programs** is the smallest set that contains $?\phi$ for each sentence $\phi$, and for which it holds that if $\pi$ and $\pi'$ are programs and $a$ an agent, then $U_a\pi$, $(\pi; \pi')$ and $(\pi \cup \pi')$ are programs as well. □

Now the truth definition given in the previous section has to be extended so that the following clause is added.

$$
l \models [\pi]\phi \text{ iff for all } m \text{ if } l[\pi]m \text{ then } m = \phi
$$

The only difficulty lies in defining the relation $[\pi]$ for updates. The other dynamic operators can get their standard interpretation.

$$
\begin{align*}
    l[?\phi]m & \text{ iff } l \models \phi \text{ and } l = m \\
    l[\pi; \pi']m & \text{ iff there is a } u \text{ such that } l[\pi]u[\pi']m \\
    l[\pi \cup \pi']m & \text{ iff } l[\pi]m \text{ or } l[\pi']m
\end{align*}
$$
3.6 Likelihoods

Again the only program that can change the information state of an agent is the update, it should not change anything else, therefore:

$$I \{U.a \pi \} m, \text{ then } I \{a \} m$$

Where $I \{a \} m$ is an abbreviation for 'I and m differ at most in the information state they assign to a.' An update should also have the same effect in PDEL as in DEL on the likelihoods or possibilities an agent considers possible. Therefore:

$$I \{U.a \pi \} m, \text{ then } L_{a,m} = \{m' | \exists l \in L_{a,l} \exists n : I[l \pi n [U.a \pi] m'] \}$$

This definition is completely analogous to the definition of updates in DEL (see page 35). But now that there is also a probability space in the information state of an agent, the probability space should also be updated. In the example with the marbles $L_{a,l}$ and $S_{a,l}$ were the same sets, and exactly the same happened to them. Those likelihoods where the marble was black were removed from these sets and the remaining likelihoods were updated. Therefore the same operation should be performed on $L_{a,l}$ and $S_{a,l}$. Thus:

$$I \{U.a \pi \} m, \text{ then } S_{a,m} = \{m' | \exists l \in S_{a,l} \exists n : I[l \pi n [U.a \pi] m'] \}$$

What happens to the $\sigma$-algebra of the probability space when an update is executed? It would seem in the example above those measurable subsets that contained a likelihood where the marble was black were removed and the remaining measurable subsets were updated. One could also say that the operation that was executed on $L_{a,l}$ and $S_{a,l}$ should also be executed on the elements of $H_{a,l}$. This is defined as follows:

$$I \{U.a \pi \} m, \text{ then } H_{a,m} = \{H' | \exists H \in H_{a,l} H' = \{m' | \exists l \in H \exists n : I[l \pi n [U.a \pi] m'] \}$$

Note that when two distinct elements in $H_{a,l}$ are updated this may result in the same new set. This happened in case of the marble example. The sets $\{k,l,m\}$ and $\{k,l\}$ were both elements of $H_{a,l}$, yet learnB results in $\{k',l'\}$ for both of these sets.

The most interesting question is how to update the probability measure $\mu$. It seems we must use conditional probability for this. In the example with the marbles the new probability, according to a, that the marble she picked was red, should equal the conditional probability that was assigned to the outcome it was red given that it was not black. In terms of likelihoods: the probability $\mu_{a,k}$ assigns to $\{k'\}$ is the probability $\mu_{a,k}$ assigned to $\{k\}$ divided by the probability $\mu_{a,k}$ assigned to $\{k,l\}$.

$$\mu_{a,k'}(\{k'\}) = \frac{\mu_{a,k}(\{k\})}{\mu_{a,k}(\{k,l\})} = \frac{\frac{1}{4}}{\frac{3}{4}} = \frac{2}{3}$$

Thus, if we want to assign a probability to an element $H'$ of $H_{a,l'}$, we somehow have to "downdate" $H'$ and $S_{a,l'}$. The new probability measure should be defined as follows:
If $l[U_a\pi][m]$, then

$$\mu_{a,m}(H') = \frac{\mu_{a,l}(H_{down}^l)}{\mu_{a,l}(S_{down}^l)}$$

where

$$H_{down}^l = \{l' \in S_{a,l} : \exists m' \in H' \exists n : l'[\pi]n[U_a\pi][m']\}$$

and

$$S_{down}^l = \{l' \in S_{a,l} : \exists m' \in S_{a,m} \exists n : l'[\pi]n[U_a\pi][m']\}$$

For this definition to work both of these downated sets must be measurable in the old situation. Again we could also give an inner measure definition in this case. Together these form the sufficient conditions for the accessibility relation of the update operator. To recapitulate:

$l[U_a\pi][m]$ iff $l[a][m]$ and

$$L_{a,m} = \{m' \mid \exists l' \in L_{a,l} \exists n : l'[\pi]n[U_a\pi][m']\}$$

$$S_{up}^{a,m} = \{m' \mid \exists l' \in S_{a,l} \exists n : l'[\pi]n[U_a\pi][m']\}$$

$$H_{up}^{a,m} = \{H' \mid \exists H \in H_{a,l}, H' = \{m' \mid \exists l' \in H \exists n : l'[\pi]n[U_a\pi][m']\}\}$$

$$\mu_{up}^{a,m}(H') = \frac{\mu_{a,l}(H_{down}^l)}{\mu_{a,l}(S_{down}^l)}$$

$$H_{down} = \{l' \in S_{a,l} : \exists m' \in H' \exists n : l'[\pi]n[U_a\pi][m']\}$$

$$S_{down} = \{l' \in S_{a,l} : \exists m' \in S_{a,m} \exists n : l'[\pi]n[U_a\pi][m']\}$$

$$\mathcal{F}_{a,m} = \begin{cases} R_{a,l} & \text{if } S_{up}^{a,m} = \emptyset \text{ or } \mu_{a,l}(S_{down}^l) = 0 \\ (S_{up}^{a,m}, H_{up}^{a,m}, \mu_{up}^{a,m}) & \text{otherwise} \end{cases}$$

This definition is clearly circular and so is the definition of likelihoods. According to Barwise and Moss this is no reason to despair, but to me it seems it is not a reason to be joyful either. To prove that a circular definition is well-defined one needs a lot of complicated set-theoretic machinery. To prove that the definition of likelihoods and the definition of updates (which is more difficult) are well-defined lies outside the scope of this thesis.

### 3.7 Conclusions and Open Questions

In this thesis I have presented three analyses of the Monty Hall Dilemma. The first employed the system for reasoning about probability in multi-agent systems. The strength of this analysis lies in the fact that it is likely to convince someone who does not believe that the player should switch doors, i.e., that Vos Savants solution is correct. The decision tree in figure 1.3 is more than enough to convince someone of this. The question is how to construct such a decision tree. In another article Halpern stresses that the protocol of a system (I would call these the rules of the quiz) is the key to making such a tree. In his article he also discusses the Monty Hall Dilemma.

What protocol describes the situation? We assume that at the first step Monty places a car behind one door and a goat behind the other two. For simplicity, let’s assume that the car is equally likely to be placed behind any door. At step 2, you choose a door. At step 3,
Monty opens a door (one with a goat behind it other than the one you chose). Finally, at step 4, you must decide if you’ll take what’s behind your door or what’s behind the other unopened door. Again, to completely specify the protocol, we have to say what Monty does if the door you choose has a car behind it (since then he can open either of the two doors). [Hal95]

He continues to discuss what I would call the lazy quiz master scenario and he also remarks that learning which door the quiz master opens does not affect the probability you assign to the car being behind a door. So all the ingredients are there for making the ‘right’ computation tree.

The weakness of this system seems to me that it is not always easy to make such a protocol. The protocol given by Halpern is not that easily extracted from the description of the Monty Hall Dilemma as it was presented to Marilyn vos Savant. Moreover it is quite informal. The strength of the analyses using DEL and PDEKS (and hopefully one day PDEL) is that one actually can specify such a protocol in the formal language itself. I would also like to stress that when one tries to solve a puzzle involving probability, the key is formalizing the rules of that puzzle. If this is done correctly, the solution can easily be found.

The weakness of the analyses with DEL and PDEKS is that the underlying set-theoretic machinery is quite complicated and will not immediately convince someone that the outcome is right. However, the ideal way to solve puzzles involving probability seems to me to simply formalize the solution in the language of PDEKS and calculate whether the solution is correct. Everything you assume can and must be made explicit. In the quotation given above Halpern says that for simplicity he assumes the car to be equally likely to be behind one of the doors. As I showed in section 3.5.2, this assumption is quite crucial for the eventual outcome, therefore it is not a matter of simplicity. This is really where the strength of PDEKS lies: every aspect of the problem can be formalised.

PDEKS is not an ideal system yet. PDEL comes much closer to this, but a lot of work still has to be put in it. First of all a proof must be given that it is well-defined. The class of likelihoods and effect of all the programs must be defined coinductively. It would also be nice to make a similar system for predicate logic. One of the advantages would be that the probability an agent assigns to a sentence could be regarded as a term. Then independence could be expressed. Another interesting extension of the language, might be adding more programs, such that more actions can be described and such that actions can be directly linked to agents (as in game theory). It would also be nice to give an axiomatization of PDEL. PDEKS must be seen as a first sketch for such a system. I hope it will prove to be a system that can deal with a wide range of problems, from the puzzles involving probability such as the Monty Hall Dilemma, to complicated formalisation of real games such as poker or chudo.

The last word about the Monty Hall Hall Dilemma, I gladly leave to Monty Hall himself. In 1975 Steve Selvin wrote two letters to the editor of The American Statistician about the Monty Hall Dilemma [Sel75a, Sel75b]. In the first letter he presented the following dialogue (Instead of ‘Monty’, Selvin writes ‘Monte,’ but in the second letter Selvin writes ‘Monty’):
Monte Hall: One of three boxes labeled $A$, $B$, and $C$ contains the keys to that new 1975 Lincoln Continental. The other two are empty. If you choose the box containing the keys, you win the car.

Contestant: Gasp!
Monte Hall: Select one of these boxes.
Contestant: I'll take box $B$.
Monte Hall: Now box $A$ and box $B$ are on the table and here is box $B$ (contestant grips box $B$ tightly). It is possible the car keys are in that box! I'll give you $100 for that box.
Contestant: No, thank you.
Monte Hall: How about $200?  Contestant: No!
Audience: No!!
Monte Hall: Remember that the probability of your box containing the keys to the car is 1/3 and the probability of your box being empty is 2/3. I'll give you $500.
Audience: No!!
Contestant: No, I think I'll keep this box.
Monte Hall: I'll do you a favor and open one of the remaining boxes on the table (he opens box $A$). It's empty! (Audience: applause). Now either box $C$ or your box $B$ contains the car keys. Since there are two boxes left, the probability of your box containing the keys is now 1/2. I'll give you $1000 cash for your box. [Sel75a]

Selvin argued that the last remark by Monty was incorrect. The player now offers to trade his box for box $C$. According to Selvin the probability that this box contains the keys is $\frac{2}{3}$. One of the letters Selvin received commenting on his first letter was by Monty Hall himself.

Monte Hall wrote and expressed that he was not 'a student of statistics problems' but 'the big hole in your argument is that once the first box is shown to be empty, the contestant cannot exchange his box.' [Sel75b]
Bibliography


