Extracting Beliefs from Hidden Protocol Situations: Additions to the Existing Formalisation

Laura van de Braak (s2165341)

12th July 2019

Master's Thesis

Artificial Intelligence
University of Groningen,
The Netherlands

Supervisor:
Prof. Dr. Rineke Verbrugge
Artificial Intelligence
University of Groningen

Abstract

In social situations, it frequently occurs that people or agents use protocols to deduce or give extra meaning to a phrase. A protocol determines the way a phrase is interpreted by the agents who hear that message. It is not always the case that all agents will be aware of the same protocols, there might be several different protocols at play. Also, the protocols are not always known to all agents, some agents may not know a specific protocol. In both of these cases, the protocols involved are referred to as hidden protocols.

An example of a situation where hidden protocols are involved is if we imagine two businessmen, one British and one American, who are attempting to plan a meeting on the first floor of a building. Unbeknownst to both of them, this phrase has a different meaning in their different cultures, so their interpretation of 'first floor' will differ: the American will arrive at the floor on ground level, where the Brit will go to the next floor up from the ground level.

Based on a formal epistemic protocol logic, introduced by Van Ditmarsch, Ghosh, Verbrugge and Wang, this project involved taking several real world examples of situations, expressing them in Epistemic Protocol Logic, and creating a simulation to show the knowledge, beliefs, and reasoning involved in a hidden protocol situation.

Epistemic Protocol Logic involves both the standard elements of a dynamic epistemic logic, and some other new ones. Among the first are elements such as actions and epistemic and factual changes. The new elements include protocols and the means to make reasoning with protocols work, such as observations and postconditions.

The epistemic protocol logic was not descriptive enough to model the nuances of reasoning deemed necessary to accurately match a real world situation, so additions to the existing formulation were postulated. The examples were then expressed in a simulation. The reasoning in the simulation was based on Kripke models, with extensions built to incorporate the complexities involved in working with reasoning about protocols.

The examples were modelled in this simulation, after which it was possible to analyse how the situations ran in a simulation. This made it possible to compare the working of the simulation to that predicted by the formalisation of the examples done earlier.

Acknowledgements

I would like to thank a few people, without which this thesis would not have been completed.

The first is my supervisor Rineke Verbrugge, whose willingness to think along and assist with any issues I encountered has been invaluable. I also want to thank Davide Grossi for the comments on my proposal and intelligent questions during my colloquium.

Furthermore, I want to thank both Annet Onnes and René Mellema for being a huge help throughout the process of creating this thesis. Whether it was to brainstorm how to turn an idea into a full thesis, to work out how some logical aspects would work together, or to help when I got stuck on specific coding issues, they were frequently available to help out. Additionally, I want to thank René for proofreading an early version of this thesis.

Next, I would like to thank all the people who have worked in the graduation rooms over the past year. From this group, I especially want to thank Kim van Prooijen and René Mellema, who were my almost daily working companions in the last few months of all our theses, and without who the last months would have been significantly less pleasant.

Finally, I want to thank my close friends and family for being there for me and supporting me through the degree and specifically the thesis.

Contents

Chapter 1

Introduction

In conversations it is often the case that people rely on extra information known to the other participants of the conversation. People do not have to describe everything in full detail. Usually we can rely on the participants of a conversation having awareness of protocols that will allow them to deduce the exact meaning. It can also occur that the protocol is known to only some of the participants, this is then known as a hidden protocol.

The concept of this project stems from the paper on hidden protocols by Van Ditmarsch et al. (2014), especially the elaboration on the concept of using protocols as a way to describe actions, observations and beliefs. Hidden protocols are intriguing to study, and there are many different types of situations where one can see their application or use. In the above-mentioned paper the authors describe several examples to illustrate the concept. Aside from the examples used in that paper, one can consider diplomatic situations, cultural differences, or even security protocols, to an extent. In these situations, it is interesting not only to consider the individual beliefs of the participants of a protocol or situation, but also to reason about to what extent they could deduce information on the beliefs of the other participants. Take one of the examples mentioned in Van Ditmarsch et al. (2014):

Example 1 (Valentine's day)**.** *Consider a café scenario with Carl, Ben and Alice. Carl and Ben are childhood friends and know each other like the back of their hands. Carl says to Ben: 'On Valentine's day I went to the pub with Mike and Sara. It was a crazy night!' This immediately catches the attention of Alice, who is in love with Mike. She asks: 'What happened?' Carl winks to Ben and says: 'Nothing'. Knowing Carl very well, Ben immediately realizes that indeed nothing has happened, whereas Alice becomes unsure of that, as she saw the wink that Carl has given to Ben (Van Ditmarsch et al., 2014, p. 19).*

The protocol in this situation is executed by Carl. It can be something along the lines of: if I make a statement hinting at something happening, and I also wink, then nothing happened'. Ben and Carl are aware of the protocol, and Alice is not. One can also assume that Ben and Carl are aware that the other is aware of the protocol, otherwise Carl would not have initiated the action that would indicate knowledge of the protocol. As for Ben, the protocol is tied to Carl being the one to initiate action, so it should be acceptable to assume that Ben is also aware that Carl is aware of the protocol. For Alice it different, she is not aware of protocols existing between Ben and Carl. However, she might have seen a similar protocol in a different context, i.e. winking to communicate to another about the falsehood of the previous statement, which would lead her to believe there is a different protocol at play.

In this small example, there are no far-reaching consequences of not correctly analysing the beliefs the different participants have of each other's knowledge of the protocols. However, in

other situations it could be of great influence to a situation.

The formalisation described by Van Ditmarsch et al. (2014) provides a logical framework for describing the effect of a protocol in a situation, given actions and observations. We believe there is some progress to be gained as regards to beliefs about protocols.

The reasoning behind this belief is that actions agents perform in these situations are chosen based on their beliefs of the situation, combined with the protocols they are aware of, and the observations they make on the current state. Assuming that observations are public, and beliefs are known, it should be possible to deduce the beliefs about protocols. The information from this beliefs about protocols can then be applied by the agents.

There will be a few different examples that can be used to show the functionality of the system on different levels. The first will be a very small one, nothing more complicated than a socio-linguistic misunderstanding e.g. "The meeting is on the first floor." This will show the effect of a simple misunderstanding, and will show that beliefs on protocols are relevant.

The second will be slightly more involved, it is example 2 from Van Ditmarsch et al. (2014), which will be introduced in Section 2.2.2 and will be explained in full in Section 3.2. This shows that in normal social situations there are cases where information can be (intentionally) hidden, and we use protocols to communicate information secretly.

Finally, there will be an example of a diplomatic situation, where the consequences of sharing or not sharing information are more complicated than in a regular friendly conversation. Also in this situation, the protocols will seem more explicit. This, together with the previous examples will show the breadth of applicability of protocols, and consequently the use of reasoning about beliefs on protocols.

In this project we take a few examples of protocol situations in which it is useful for the participants to know what protocols the other participants are aware of. These examples will then be expressed in the existing formalisation and show that there is a part – the beliefs-on-protocols – that cannot be expressed properly. Additionally, a simulation similar to the formalisation will be created, based on some pseudo-logic, that can show the same.

The result of this project will be the addition of a semi-formal description of additions needed to make reasoning about the belief on protocols of other agents possible to the existing formalisation. It will include theory and modelling due to studying the existing formalisation and applying it to situations. It includes implementation due to the simulation aspect. Furthermore, the inclusion of several examples of a different type should show that my assumptions will hold on more than one specific example. Finally, the ideas of the situations mentioned are ones of which relevance can be shown.

The exact research question will be as follows: Is it possible to extend the existing formalisation so that we can represent real world examples? Can we show this in a simulation?

The next chapter will give relevant background knowledge. After that will be a chapter discussing the exact examples that will be used. Chapter 4 formally introduces the theoretical framework created by Van Ditmarsch et al. (2014). Next, Chapter 5 will show the examples expressed in the formalisation. After this, there is the Chapter 6 discussing the design of the simulation, followed by Chapter 7, which shows the working of the examples in the simulation. Finally, Chapter 8 will discuss how well the theoretical formalisation and the simulation work, and Chapter 9 will conclude the thesis with conclusions and future work.

Chapter 2

Relevant Literature

This project has a few core elements that are tied together. This chapter will cover the different theoretical aspects that are to be combined, as well as provide an overview of relevant prior research.

First, in Section 2.1 the concept of protocols will be introduced and explained. Then there will be a discussion on hiding and hidden information in Section 2.2. Section 2.3 will be on formal logic, including extensions to Dynamic Epistemic Logic that seem useful to this project. Finally, Section 2.4 will touch on simulations, and how they can be used in conjunction with formal reasoning.

2.1 Protocols

Protocols can be used in a variety of situations and circumstances. A protocol (in conversation) occurs any time when the meaning of a sentence, phrase or statement is not the literal meaning of the words. A use for protocols on a larger scale that immediately springs to mind is protocols of the security protocols type (Teepe, 2006). These protocols are agreements made by multiple parties that describe how to transfer information securely. Most of these protocols are aimed at performing transfers of information during which no unauthorised or unintended parties can gain access to the information being transferred. This section will touch on some of the research done on protocols, as well as how they can be used.

2.1.1 Wittgenstein's Language Games

Ludwig Wittgenstein was the first to describe language in terms of a protocol, though he described it by a different name. His concept of language games was introduced in the post-mortem publication of his book *Philosophical Investigations*. He said that the meaning of a word was a result of applying the rule of the game that was being played. The game determines which rule is most important (Wittgenstein, 1953). In the terminology as used in this thesis: the context of a situation determines which protocol is at play.

The concept of language games was broadly used by Wittgenstein to refer to different types of these games. One is the concept of gaining meaning through the application of rules on a word level. For example, the word 'Apple' can be used in different ways, depending on the context. It can be as response to a question, e.g. what fruit someone wants to eat. It can be a request, asking someone to bring you an apple. It can even be a warning to watch out for the apple falling

out of the tree someone is standing under. The rules can also be applied on a sentence level, to refer to the meaning of a phrase.

The analogy can be made between language and games in general that the rules of the game determine the meaning of an action played. Though on a more elaborate level, Wittgenstein claimed that language has similar rules that determine the meaning in general conversation.

2.1.2 Grice's Theory of Implicature

Following Wittgenstein, others also looked into understanding semantics of language. In other words, they want to be able to show how we know what language *means*. Grice discusses cooperative conversation and the meaning and implicature of language. Especially his analyses of implicature are interesting.

Implicature refers to the meaning of a sentence or utterance, even when that meaning is not necessarily directly stated or implied. An example of this can be found in the following situation. Take an office with a few people. Around lunch-time one of them asks his co-workers if they want to go for lunch. One of them responds with "I could eat something". When looking at just the literal meaning of the sentence, the meaning is that the speaker possesses the ability to eat something. Conversationally, and connected to the context of the situation, there is the implicature that the speaker is saying that they would like to go for lunch.

Grice's theories could help with analysing the implicatures of the utterances in the examples used in this project. He differentiates between conventional and non-conventional implicature. The conventional implicature of an utterance is the one that is the literal meaning of an utterance. Non-conventional implicature is frequently a conversational one, so one where the meaning is implicated not just by the utterance itself, but by the context provided by the rest of the conversation.

Grice also coined maxims for cooperative communication. These are a set of norms people are expected to follow to ensure a cooperative communication. There are four: the maxim of quality, the maxim of quantity, the maxim of relation, and the maxim of manner. The maxim of quality means that people are expected to tell the truth, or the truth as they know it. The maxim of quantity relates to the fact that people should provide enough information to be understood, no more and no less. The maxim of relation entails that what people say has to be relevant to the topic of discussion. Finally, the maxim of manner states that people should avoid ambiguity and obscurity, and should instead be direct and straightforward.

Something to remember when looking at Grice's theories is that in his examples the interactions are generally cooperatively intended. In other contexts this is not always the case, as on some occasions people or agents will have a competitive aim, or even a malicious aim, and those should not be left out of models. That would result in incomplete models.

2.1.3 Parikh and Ramanujam's Framework

Parikh and Ramanujam take Grice's theory of implicature into account in their framework. To better analyse the semantics of messages, they limit themselves to distributed systems of computing agents that can only communicate through passing messages. How to determine the intended meaning of a message can depend on the medium of the message, on the state the receiver is in, what state the rest of the system is in, and more factors.

They first discuss different types of interpreting or limiting protocols: intensional and extensional. Intensional is that which syntactically and semantically exactly fits in the protocol description. Extensional is all the possible evolutions of interpretation after that. The description of protocols often becomes circular, but knowledge-based specification of protocols is said

to help. Messages are sent in order to transfer information. The system operates under the assumption that each agent only has a partial view of the system state, so each message will increase the knowledge of the agent.

There are plenty of issues in resolving interpretation; the solution they decided on is that states are specified by properties stated in a logical language. The view of an agent at a state consists of the global properties which are *known* to the agent at that state. Communication is denoted as a set of pairs (α, β) such that if at state s, α is known to the agent, then β will be known in the next state.

In their model they make use of global and local histories. Global histories represent possible system evolutions given by sequences of global events. Local histories are created by projecting global histories on to local components. Additionally, Parikh and Ramanujam define *timelessness* and *persistence*. A message A is timeless if its truth value does not depend on the time, and persistent if after time k it remains true whenever it is true at k. What options an agent has (for actions in the next time step) and how it chooses depends on the protocol, and the utility and informativeness of a message.

An intensional description of protocols is one that specifies, for each agent, at any finite global history, what global event that agent may participate in. The extensional protocol defines the possibilities for the global history. The circularity comes into play because the protocol defines when messages can be sent, and what to do on receipt of a message, but the meaning of the message depends on the protocol used. This can be resolved by considering only extensional protocols that can be realised by intensional protocols as models of knowledge formulas. Or as they put it: "if a knowledge formula has an extensional protocol as model, then there is an intensional protocol which realises it"(Parikh and Ramanujam, 2003). In their formalisation they then add a message semantics in which the meaning of message m is based on both the state of the agent and the local history of the agent.

The relevance of this to the current project is to consider the implications of messages, as their implicature may contain more information than the conventional interpretation would have. This implicature depends on the local history, known protocols and the agents' own perceptions.

2.1.4 Wang's Framework

The PhD thesis by Wang contains some useful ideas, though his application is a bit different from the aim of this project (Wang, 2010). He uses public announcements as the only communication method, and thus avoids the problem of dealing with difference between perception and announcements. He says that: "An *explicit* set of sequences of events is an *extensional* notion of *common sense* protocols which are usually specified by a few rules governing the communications." He goes on to mention that any decent protocol should specify what the agents need to do in any possible situation.

In a protocol, as he defines it, there are three separate elements. The protocol needs to be expressly separate from the initial assumptions of the protocol (or in other terminology: the preconditions). Additionally, the intended goal of the protocol is stated explicitly. In this system it is desirable for a protocol to be deterministic. This does not mean that syntactically it has to be deterministic, more that in practice there cannot be more than one outcome *after* it has started being executed.

An aspect this author feels very strongly about is that meta-knowledge of a protocol can affect the verification of a protocol. Meta-knowledge can have an influence on the situation at hand. We show reasoning about protocols in the formalisation of examples in Chapter 5.

In Wang's framework, all protocols are public. This causes the agents within the framework

to assume that all other agents will not take actions that do not lead to the goal. This ties in with a rationality assumption, one that is not uncommon in logical puzzles or models.

2.1.5 Van Ditmarsch, Ghosh, Verbrugge and Wang's Framework

The paper that was the core inspiration for this thesis is Van Ditmarsch et al. (2014). They set up a formalisation for protocols in formal logic, the details of which will follow in Chapter 4. They briefly discuss protocols, and that there are many ways to define when an agent knows a protocol, before defining it as "understanding the underlying meaning of the actions induced by the protocol." After that, most of the paper is devoted to setting up a new logic, Epistemic Protocol Logic, that can deal with both epistemic and factual changes to the state after application of protocols. They end their paper with applying their logic to an example, to show how a protocol could be defined for them, which could then be used to analyse the protocol.

In their descriptions, agents can be aware of protocols, and will define the information they gather from an announcement or observation. Not all agents have knowledge of the same protocols. Observations are public: they are the representation of the effect an action has on the environment, comparable to how an announcement is a representation of a purely epistemic change. Protocols will describe the epistemic updates an agent should make internally after the preconditions have been met (when certain observations have been made). If some agents do not know of certain protocols, their beliefs after an observation or announcement may differ from those of other agents in the situation.

Following this definition of protocols, they can be anything from security protocols to normal conversations or calculated solutions to logic puzzles. It is this breadth of possible applicability that is used in this project.

2.1.6 Application Possibilities

In this thesis, protocols will be able to describe any of the implied situations. Mostly, the interpretation from Van Ditmarsch et al. (2014) will be used. This allows for protocols to be used to describe beliefs an agent has on the impact of actions on a situation. These can be seen as similar to if-then rules that describe: if A happens, then B is the result. In this situation, A can refer both to actions or announcements, and B can be anything from changes in belief to changes in the world. These changes in belief are not limited to one agent, but could be within any agent aware of the protocol. A special case is when agents that are unaware of the actual protocol can have an internal belief of a protocol that will let them believe something else than the truth. These internal protocols can come from just the assumptions the agent works under, and do not necessarily lead them to true beliefs about the world they are in. It is situations like these that are interesting to study and that will be considered in this project.

2.2 Hiding and Hidden Information

There are many different reasons for hiding information, and many different ways that information can be hidden. In this section this will be discussed on a conceptual level, with an emphasis on the useful elements for this project, both on a practical and on a conceptual level.

2.2.1 Keeping Information Secret

Concealing information is as old as civilisation. The reasons for concealing information are as plentiful as the ways people have found to conceal information (Singh, 1999).

One reason is for privacy. If one is sharing personal information with someone else, that does not mean that they are also fine with having others know this information. A clear example for this is from Van Ditmarsch et al. (2014).

Example 2 (The voice of Kathleen Ferrier)**.** *Consider a café in the 1950s, with three persons, Kate, Jane and Anne sitting across a table. Suppose Kate is gay and wants to know whether either of the other two is gay. She wants to convey the right information to the right person, without the other getting any idea of the information that is being communicated. She states: 'I am musical, I like Kathleen Ferrier's voice'. Jane, who is gay herself, immediately realizes that Kate is gay, whereas, for Anne, the statement just conveys a particular taste in music. (Van Ditmarsch et al., 2014, p. 18-19).*

Even though Kate wants to share her personal information - that she is gay - to others who are gay, she does not want just anyone to know. Motivated by the desire to keep her private information private, she hides the information from others not in the know.

Another possible motivation for keeping information secret is that, while in conflict with other parties, you do not wish them to know what you are planning. Any war situation is a good example for this. If a general is planning assault on a castle, and needs to communicate with troops about when to attack, they do not want the enemy to be able to understand when the attack is coming. If they do not know the exact time of attack, they might be under-prepared, which would be fortunate for the attacking general and his armies.

This last type of situation can be turned around to make it align more closely with another reason: protection. If there is suspicion that people are planning to attempt to assassinate the general, it is crucial that the general is protected. This can be done by keeping guards on the general, but also by hiding what the general's exact plans are. If the assassins do not have the exact information on where the general will be, it is more difficult for them to prepare an assault.

2.2.2 Hiding Information in Information Transfers

An example of hidden information stems from the ancient Greek historian Herodotos. He wrote about the history of the Greeks and the peoples they interacted with. One of these hidden information actions was in the build up to the war with the Persians. This example comes from Singh (1999).

Example 3 (Preparing for a Persian Invasion)**.** *A Greek living in Persia had noticed the war preparations, and endeavoured to send a secret message to the Spartans to warn them of the impending invasion. As the danger of discovery was great, there was only one way in which he could contrive to get the message through: this was by scraping the wax of a pair of wooden folding tablets, writing on the wood underneath what Xerxes intended to do, and then covering the message over with wax again. In this way the tablets, being apparently blank, would cause no trouble with the guards along the road. When the message reached its destination, no one was able to guess the secret, until Cleomenes' daughter Gorgo, who was the wife of Leonidas, the ruler of the Spartans, divined and told the others that if they scraped the wax off, they would find something written on the wood underneath. This was done; the message was revealed and read, and afterwards passed on to the other Greeks.*

Because the Greeks were warned by this hidden message, they managed to be prepared when the invasion came.

To communicate while hiding information, one needs to ensure that the hidden information cannot be extracted from a communication by everyone, but can still be understood by the intended recipient. To do this, a sender of a message needs to have an understanding with the receiver about the full meaning of the content of a message.

This understanding comes in the form of a protocol: 'if this type of message is sent, then the sender intended the following meaning to be understood'. With information being so ubiquitous, these protocols have become more pronounced. They are used everywhere, both for relatively low-level interactions, and in security protocols. Security protocols are created because it is desired to be able to transfer information without risk of interception. Or in the case that a message is intercepted, that the intercepting agent cannot understand the message transmitted.

This is the type of hiding information that this project will focus on. There is some related research that gives a few different angles to consider, both regarding what information to hide, as well as how to hide it.

2.2.3 Lying and Suspicions in Communication or Strategy Games

In creating a set-up for Dynamic Epistemic Logic, Baltag; Baltag et al. took complicated interactions into account (Baltag, 2002; Baltag et al., 1998). These situations are things like: having secure group announcements, group announcements with a suspicious outsider, group announcements with a secure wire tap, and they can also include performing misleading epistemic actions and experiencing paranoia (Baltag et al., 1998).

Baltag (2002) mentions four different types of epistemic changes. The first is through direct information gathering. This is information obtained through an agent's own perception. The second is the communication of information, such as through sending and receiving messages, public announcements, and the interception of messages. Next is information-hiding: secret communication, lying, and sending encrypted messages. Finally, there is information-loss, or misinformation, gained through being deceived, or holding incorrect beliefs.

This can be seen in Van Benthem et al. (2006) & Van Eijck (2012). In defining their logic for change and communication, they briefly mention that their system can also deal with the complexity in secret group communication and common belief.

Another interesting point of view is as presented in Ågotnes et al. (2018). The authors discuss lying announcements: announcements that are false until spoken, when they become true. They call these announcements 'true lies'.

True lies are something that is not allowed to exist commonly in public announcement logic, which usually makes the assumption that all announcements are truthful. It is a complex situation, because while at the time of announcing the statement it may be false, when it has spoken it has changed to true. The announcement itself has caused factual (or epistemic) change. This is an example of what is known as an unsuccessful update. A successful formula, in public announcement logic, is one where $[\varphi]\varphi$. In the case of a true lie, the following holds: $[\neg\varphi]\varphi$.

2.2.4 Hiding Information

A different perspective comes from Grossi et al. (2016). In this paper, the researchers study situations in which part of the communications is visible to all agents, and another part is not. In their specific example all computations are visible – agents can see that a switch is flipped – but the data is only visible to agents with access. Only agents that can see that switch know whether it is flipped to on or to off. This is in contrast to what was mentioned in Sections 2.2.1 and 2.2.2, where the communication itself is kept secret, and the information contained within

the communication is so as well. This shows the variety of ways in which information can be hidden.

2.2.5 Hiding Information from a Security Perspective

As mentioned before, being able to hide, protect or secure information has become increasingly important as we properly entered the information age (Teepe, 2006). The information we share online is both more personal and more easily accessible to complete outsiders than it used to be. Due to this, it is key that information is protected. Computer security focusses on protecting information from unwanted eyes. There are many different techniques and protocols that can be used for this.

One of these is encryption. In concept it is similar to how a Caesar cipher[1] would encode the plaintext of a message, so it could not be read. The receiver of a message would know the decryption key (the shift of the cipher), and manage to translate the encrypted message back to its original state.

This is an example of symmetric encryption: the encryption and decryption keys are the same. Another possibility is to use asymmetric encryption, where the encryption key differs from the decryption key. Generally, in such cases, one of these is public, where the other is kept private and secret.

A message can be sent to an agent using their public key to encrypt, and only with the private key of that agent will it be possible to decrypt this. Another option is for an agent to encode a message with their private key, and anybody with their public key will be able to decode the message. This last option is not used for keeping messages secret, but for adding a signature to the message to prove that the sender is who they say they are.

2.3 Logic

One of the methods used for studying reasoning is through formal logic. Most commonly used is Dynamic Epistemic Logic (DEL). Dynamic Logic is a type of formal logic used for cases where the situation changes at some point (Van Ditmarsch et al., 2007). Because of this, it is useful when dealing with multi agent situations. Because of the dynamic aspect of the reasoning, it is possible to manage situations in which multiple agents interact with the environment – or each other – simultaneously. Dynamic Logic can also model in a way that seems closer to programming than to formal logic. Instead of only working with adding propositions to a database, it can test if propositions are true, and deal with statements such as 'after performance of φ, ψ is true'. In this statement, φ can be a complex set of actions and statements. This is the grounding we need to reason about protocol situations.

Another commonly used type of logic for this type of reasoning is Epistemic Temporal Logic (Fagin et al., 2003). In this, instead of using the dynamic structure shown later, a temporal operator is added to Epistemic Logic.

2.3.1 Belief and Knowledge in Logic

Aside from just being able to reason about the state of ψ after running of φ, we wish to make it possible to reason about the knowledge agents have of the situation. This is also where the

[1] The Caesar cipher is one of the earliest well known ciphers. The plaintext is encoded by 'shifting' each letter a fixed number of positions down the alphabet. The standard Caesar shift was a left shift of 3, so each 'a' is encoded as an 'x', a 'b' as a 'y' etc. This message is then decoded by shifting the letters with a right shift of 3 to their original state.

'epistemic' in Dynamic Epistemic Logic stems from. DEL is the logic that is used not only to represent a dynamic environment, but also to incorporate reasoning on knowledge within that environment.

Because in this project there will be more emphasis on beliefs than on knowledge, we will briefly elaborate on the differences between reasoning about belief and reasoning about knowledge (Van Ditmarsch et al., 2007). When reasoning about agent situations, it is often desirable to be able to reason about not just the facts in a world, but also the knowledge and beliefs the agents hold about the facts in the world and about each other's knowledge.

Reasoning about knowledge is more definite than reasoning about belief, and is desirable if you want proof of something. Reasoning about belief is not as rigid, but on the other hand, it is also less informative. In this project the focus will be reasoning using belief, because the agents cannot know the internal states of other agents, but they can form beliefs about them. Reasoning about knowledge is written as $K_i\varphi$: agent i knows that φ. For belief it is similar: $B_i\varphi$: agent i believes that φ.

When reasoning about beliefs, one can speak of Doxastic Logic. In Doxastic Logic it is possible to use axioms to define the levels of rationality of an agent. However, it is more common to see the belief operator added in to Dynamic Epistemic Logic (Meyer and van der Hoek, 2004).

2.3.2 Group Logics

Often in reasoning, terms such as common knowledge or common belief will be used. This project will also use group information, and the difference between separate agents knowing something and groups knowing (or believing) something will be important.

In groups there can be reasoning on individual agents. In the same vein, we can reason about groups of agents. It is possible to reason about a specific group of agents, and state that all the agents in group G know that φ.

The more interesting forms of reasoning are reasoning about General and Common Knowledge. General Knowledge is when all agents know something. It is usually written as $E\varphi$: every agent knows that φ. There is also a version where not only do all agents know φ, but all agents know that all agents know φ, and all agents know that all agents know that all agents know φ, and so on ad infinitum. This is common knowledge, and written as $C\varphi$: it is common knowledge that φ.

Common Knowledge and General Knowledge can also be stated for specific groups as follows $C_G\varphi$ means that between the agents of group G, it is common knowledge that φ (Baltag et al., 1998; Van Benthem et al., 2006).

2.3.3 Announcements in DEL

Public announcement logic is the extension of epistemic logic that makes use of the concept of programs (from Dynamic Logic) to be able to signify how agents can pass information along to other agents. In the protocol situations this project expects to encounter, announcements or statements are made and will influence the belief states of the separate agents (Baltag et al., 1998; Van Ditmarsch et al., 2007).

An announcement is written as follows: $[\varphi]\psi$, which means that after announcement of φ, ψ holds. This announcement can be done by an agent in the environment, or can come externally. If an announcement is public, then afterwards the information contained in the statement becomes common knowledge.

Contrary to what one might expect, it is not always the case that $[\varphi]\varphi$. For example, take for φ: "You don't know this, but it is raining in London". The statement contains two facts. The

first: 'you do not know that it is raining in London', and the second 'it is raining in London'. Of course, after the announcement, you know that it is raining in London, so the first part of the statement becomes false, leaving φ to be false after announcement of φ. This is called an unsuccessful update, and is important to keep in mind when constructing the protocol logic later (Van Ditmarsch et al., 2007).

DEL also has private and secret announcements. A private announcement is when an announcement is made to a single agent or group of agents, and the other agents cannot hear what was said. They can, however see this happening, so can reason about the message being passed along. Secret announcements are also made privately. The difference between private and secret announcements is that with secret announcements, other agents cannot see that an announcement has happened, so they will not be able to reason about this announcement.

2.3.4 Protocol Logic

Protocols can be described through programs. These programs work by cases. The actions taken depend on the cases met. Or in other words, if the preconditions for a specific action are met, that action will be performed. We wish to formally express the state of the world after certain observations, with and without knowledge of the protocols at play. Some of the agents will be able to deduce the implicature of the announcements and observations, and some will not. This difference is dependent on whether the agent has knowledge of the protocol that the initiator of the announcement or observation is following (Fagin et al., 2003, Chapter 5). For these we need a formalisation that contains these separate elements. We need to be able to express observations, announcements, and protocols. We also need to be able to discuss knowledge and/or belief, and group logics. The formalisation by Van Ditmarsch et al. (2014) contains these elements.

2.3.5 Temporal Logic

Temporal Logic is a form of modal logic, with the modal operators denoting different paths through time (Meyer and van der Hoek, 2004). These operators will differ per specific type of temporal logic, and there are several types. The modalities intended can be 'always', 'until', 'next', 'eventually' or others.

One specific version of temporal logic is Linear-Time temporal logic, which operates under the assumption that time is linear and discrete, and only linear actions are possible (Huth and Ryan, 2004, Section 3.2). Modal operators here include $X\varphi, F\varphi, G\varphi, \varphi U\psi,$ and, $\varphi R\psi$. X means 'φ holds in the next state', F means 'φ holds in some future state', G means 'φ holds in all future states (globally)', $\varphi U\psi$ means 'φ holds until ψ is true'. $\varphi R\psi$ (release) means that 'ψ has to be true up to and including the point where φ becomes true'.

Most of these modalities will not be used in this thesis, but the concept of drawing linear paths through time and expressing them formally is one that will be returned to.

It may seem that such a temporal logic is not easily combinable with the epistemic and protocol-related concepts the rest of this logic section uses. This is not completely true, as research by van Benthem et al. (2009) shows that Epistemic Temporal Logic (ETL) and Dynamic Epistemic Logic can be combined. Epistemic Temporal Logic shows how knowledge changes through time as the information available changes.

Their combination is created by reformulating both ETL and DEL in protocol structures, which can translate into each other. This is not an aspect actively used in this thesis, but it is relevant to know that temporal logics can be translated into DEL structures and vice versa, if we want to combine information of both types.

2.3.6 Semantics of Public Announcement Logic with Common Knowledge

The semantics description in this section originates from Van Ditmarsch et al. (2007, Definition 4.7). This section covers what the semantics of the above-mentioned extensions to epistemic logic are, both formally and descriptively. The next section will contain an example in which this definition will be referred to, to illustrate the information shared here. The basic statement in public announcement logic is $[\varphi]\psi$. This means that after announcement of φ, ψ holds, so the worlds in which ψ is not true after announcement of φ will be cut off from the other worlds. It will not be reachable through the accessibility relations.

Definition 1 (Semantics of Announcement Logic)**.** For agents A and atoms P, an epistemic model $M = \langle S, \sim, V\rangle$. S is the set of states, $\sim$ the relations between states and V the valuations of the atomic propositions. With a an arbitrary agent, B an arbitrary group of agents, and p an arbitrary atom, the semantics of announcement logic are defined as follows:

$$
\begin{array}{lll}
M, s \vDash p & \text{iff} & s \in V_p \\
M, s \vDash \neg\varphi & \text{iff} & M, s \nvDash \varphi \\
M, s \vDash \varphi \wedge \psi & \text{iff} & M, s \vDash \varphi \text{ and } M, s \vDash \psi \\
M, s \vDash K_a\varphi & \text{iff} & \text{for all } t \in S : s \sim_a t \text{ implies } M, t \vDash \varphi \\
M, s \vDash C_B\varphi & \text{iff} & \text{for all } t \in S : s \sim_B t \text{ implies } M, t \vDash \varphi \\
M, s \vDash [\varphi]\psi & \text{iff} & M, s \vDash \varphi \text{ implies } M|\varphi, s \vDash \psi
\end{array}
$$

with $\sim_B$ defined as: $(\bigcup_{a\in B} \sim_a)^*$.

Taking $[\![\varphi]\!]_M$ for $\{s \in \mathcal{D}(M) \mid M, s \vDash \varphi\}$, we can define $M|\varphi = \langle S', \sim', V'\rangle$ as:

$$
\begin{aligned}
S' &= [\![\varphi]\!]_M \\
\sim'_a &= \sim_a \cap ([\![\varphi]\!]_M \times [\![\varphi]\!]_M) \\
V'_p &= V_p \cap [\![\varphi]\!]_M
\end{aligned}
$$

The abbreviations $\vee, \rightarrow,$ and $\leftrightarrow$ are defined in the usual manner. $\hat{K}\varphi$ is defined as $\neg K\neg\varphi$.

2.3.7 Example in Public Announcement Logic

The best way to show the extensions common knowledge and public announcement logic to DEL is through an example. The example we chose is taken from Van Ditmarsch et al. (2007, chapter 4), and they either call it 'the three cards game', or 'Hexa'. Because this game features common knowledge, announcements, and general reasoning about knowledge, this is a good example to show how these features can work together.

In the three cards game there are three agents: Anne, Bill, and Cath, and three cards: 0, 1, and 2. Each agent has one card. Notation for 'Anne holds card 0' is 0_a. The goal for the agents is to find out the card distribution among the agents. Each agent is aware of their own card, but cannot see the others' cards. This example contains common knowledge, public announcements, and knowledge. The true situation is that Anne holds card 0, Bill holds card 1 and Cath holds card 2: $0_a \wedge 1_b \wedge 2_c$. We will use the shorthand 012 for this. Agents can make announcements, which is how they communicate information. They can use this to gain information. Anne can announce which card she is holding to the other players.

In this case knowledge is formulated as $K_a 0_a$. Anne knows that she holds card 0. We occasionally also make use of the notation $\hat{K}_i\varphi$, which means that agent i considers φ to be possible.

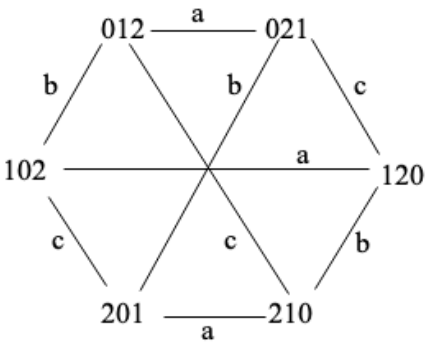

Figure 2.1: *Kripke model of Hexa when every agent has looked at their own card, and no announcements have been made yet.*

In some other systems it can be notated as M (for Maybe), but we found $\hat{K}$ to be less confusing. The Kripke model of the agents' knowledge can be seen in Figure 2.1 It is common knowledge that there are three cards that are distributed over the three agents, and each agent holds exactly one card. For common knowledge among all agents within the game, we use C with no subscripts. So the common knowledge that each agent holds one card and that there are three different cards to be divided can be described as follows: $C((0_a1_b2_c) \vee (0_a1_c2_b) \vee (0_b1_a2_c) \vee (0_b1_c2_a) \vee (0_c1_a2_b) \vee (0_c1_b2_a))$.

In the above situation, Anne announces that she holds card 0. This is notated as $[0_a]_a$ to model the announcer of the announcement. Because of the assumption that announcements must be true at the time they are uttered, the effect of this announcement is that Bill and Cath know that Anne holds card 0: $[0_a]_a(K_b0_a \wedge K_c0_a)$.

After that announcement, the knowledge of all agents changes. For example, it is now knowledge (and common knowledge) that Anne holds card 0 ($C0_a$).

Because Bill and Cath both hold one card themselves, and they know that Anne has card 0, and they know that each agent has one card, they both know the distribution of the cards is $0_a1_b2_c$. Formally:

$$K_b(1_b \wedge [(0_a1_b2_c) \vee (0_a2_b1_c) \vee (1_a0_b2_c) \vee (2_a0_b1_c) \vee (1_a2_b0_c) \vee (2_a1_b0_c)] \wedge 0_a),$$
$$\text{therefore } K_b(0_a1_b2_c)$$

Similarly:

$$K_c(2_c \wedge [(0_a1_b2_c) \vee (0_a2_b1_c) \vee (1_a0_b2_c) \vee (2_a0_b1_c) \vee (1_a2_b0_c) \vee (2_a1_b0_c)] \wedge 0_a),$$
$$\text{therefore } K_c(0_a1_b2_c)$$

On the other hand, Anne only knows that the other players know her card, but still does not know the distribution. So unless Bill or Cath makes an announcement of their own card, Anne is stuck with not knowing.

Looking at this from the semantic angle as in Definition 1, this means that the worlds in which Cath and Bill do not know the distribution of cards are not considered possible any more. There are no accessibility relations left between the true world and the worlds in which the distribution is unknown to every agent.

A different option for Anne is to announce that she does not have a specific card. So, Anne can say she does not have card 1. This does not tell Bill anything, as Bill has card 1 himself and therefore Anne cannot have it. It does give Cath enough information to deduce the distribution of cards. Bill considers cards 0 and 2 possible for Anne.

$$
\begin{aligned}
&[\neg 1_a]_a C \neg 1_a. \\
&\qquad \text{therefore } K_a(0_a \wedge K_{bc} \neg 1_a), \\
&\qquad \text{therefore } K_b(1_b \wedge \neg 1_a) \Rightarrow K_b 1_b \wedge (\hat{K}_b 0_a \wedge \hat{K}_b 2_a), \\
&\qquad \text{therefore } K_c(2_c \wedge \neg 1_a) \Rightarrow K_c(2_c \wedge 0_a) \Rightarrow K_c(0_a \wedge 1_b \wedge 2_c)
\end{aligned}
$$

Anne still does not have the information necessary, but if Bill announces that he still does not know which card she has, that gives Anne enough information to know the distribution. If Bill does not know her card, that means her announcement did not give him enough information. The only way that could happen is if he himself has card 1. Given that Anne knows that she has card 0, and now that Bill must have 1, she knows the distribution. Additionally, Anne can now reason that Cath must also know the distribution.

$$
\begin{aligned}
&[\neg 1_a]_a \neg(K_b 0_a \vee K_b 1_a \vee K_b 2_a), \\
&\qquad \text{therefore } [\neg(K_b 0_a \vee K_b 1_a \vee K_b 2_a)]_b K_a 1_b, \\
&\qquad \text{therefore } K_a(0_a 1_b 2_c) \\
&\qquad \text{therefore } K_a K_c(0_a 1_b 2_c)
\end{aligned}
$$

This same result could have been reached if Cath had said that she now knew what card Anne had. The only way Cath would know that is if she did not have card 1 either. Because there are only three cards in the system, and Anne has card 0, Cath must have card 2.

$$
\begin{aligned}
&[\neg 1_a]_a K_c 0_1, \\
&\qquad \text{therefore } [K_c 0_a \vee K_c 1_a \vee K_c 2_a]_c K_a 2_c, \\
&\qquad \text{therefore } K_a(0_a 1_b 2_c), \\
&\qquad \text{therefore } K_a K_c(0_a 1_b 2_c)
\end{aligned}
$$

In all these cases, the agents come to their separate conclusions by removing from their set of possible worlds the ones which are no longer accessible following application of Definition 1 after the information update caused by the announcement.

2.4 Simulation

Simulations on reasoning can be used in several cases. Simulations seem to be used for either model checking or verification, or when testing multiple options for variables, or in a game situation where the players determine the outcome of the situation. A lot of the more game-theoretical papers have simulations, just as any other situation where strategy comes into play.

2.4.1 The Benefits of Simulations

For the simulation of this project, we will make a representation of the framework presented in Van Ditmarsch et al. (2014).

We will use the simulation both as the means to an end as well as a goal in and of itself. On one hand it will be a tool to show what we believe is missing from the current framework. On the other hand it is a goal because it is then usable in situations.

The use of a simulation is beneficial in these types of examples, because the formal logic expression of the framework could potentially be too rigid a framework to express the full complexity of a situation. The simulation would allow for more flexibility which would allow for experimentation.

Another benefit of a simulation is that there are less calculations to perform by hand if there are similar situations or longer runs of the same actions. One idea is to have different 'parameter settings' within simulations. This would be to change the initial state of the situation, and see how it plays out. This can then be used to show the effects of differences in beliefs or responses. Another option is to have some way of allowing for more and less flexible representation of situations.

2.4.2 Examples of Simulation Types

There are diverse examples of models or situations that are simulated. Grüne-Yanoff and Weirich (2010) provides an overview of different types of occasions simulations could be used, and what relevant addition they could provide. In doing so, they discuss the difference between computing something and creating a simulation, and the difference between simulating an equation versus simulating an agent-based system. The agent-based systems are particularly well-matched to creating simulations of social situations, as was also discussed in Zoethout and Jager (2009).

Simulations are often used to model intelligent behaviour in agents (Wijermans et al., 2008; Dykstra et al., 2013, 2015; De Weerd et al., 2013, 2015, 2017). In Wijermans et al. (2008) simulations are used to model goal-driven behaviour. The authors use an example of agents at a festival location. Agents can have multiple goals, and the intricate part of the simulation was focussed on dealing with prioritising those goals, and then using the simulation to show how the group of agents would respond over time.

Dykstra et al. built a model on group dynamics in discussions or debates. The authors created a simulation in which agents had a reputation score. Whether an agent would share their opinion in a debate was dependent on their own reputation score and the overall opinion of the agents around them. Over time, this showed how groups of agents with the same opinion form.

De Weerd et al. studied different hypotheses behind the evolution of theory of mind. To test these hypotheses, the authors modelled agent simulations for different games. These games were used because they could be used to test the hypothesis. They had a competitive game, a cooperative one, and a mixed-motive game (De Weerd et al., 2013, 2015, 2017). The agents in the simulation where created to be able to deal with different order theory of mind. In the games, it would then be tested which order theory of mind was most beneficial to use. This information was then used to discuss the veracity of the hypotheses.

In this case, a simulation was created so the steps of playing a game could be repeated arbitrarily, without needing to work them all out manually. The authors also performed experiments where the computational agents would play against human agents. This was done to compare that performance to the one with only computational agents, and could be seen as verification of the agent models.

In Verbrugge (2009), the author describes the connection between logic and (higher-order) social cognition. In doing so, she touches on a few facets of reasoning that can be simulated. Comparatively, Gaudou et al. (2011) do almost the exact opposite of what this project is. They take a well-known social simulation, and show that it can be expressed in logic. The logic they use is a dynamic logic of propositional assignments, and is formulated in such a way that it is closely related to the simulation they are translating.

Chapter 3

Example Situations

This chapter will describe the examples used in this project. These examples will return during the rest of this thesis in Chapters 5 and 7 when they are being described formally, or placed in a simulation. The examples increase in complexity of the amount of reasoning that goes into them and the possibilities of analysis. Additionally, the consequences of not having the hidden knowledge increase per example. That is not quantifiable, as the consequences cannot retain their meaning when taken out of context, but should still be apparent while reading them.

3.1 Language Example

This example is of a socio-linguistic misunderstanding. This section will contain a description of the situation, followed by an analysis of the reasoning that happens within the situation.

3.1.1 The Situation

The following example was inspired by Carston (1999).

Example 4 (A socio-linguistic misunderstanding)**.** *Take two businessmen, a Brit and an American, who want to agree on a location for their meeting. The location is 'on the first floor'. Unbeknownst to the businessmen, they have a different meaning attached to that sentence. The concept of 'the first floor' of a building is culturally ambiguous. For American English speakers, they use this to refer to the ground floor - the floor on ground level. British English speakers will say this when referring to the next floor up from the ground floor. When the time for the meeting arrives, the businessmen will find themselves on different floors.*

A dictionary entry for 'first floor' gives the definition as shown in Figure 3.1. This entry is from Merriam Webster, one of the larger American dictionaries[1].

The cultural ambiguity allows for misunderstandings about the shared information based on personal protocols, which is why we use this as an example. It is an example that has a small reasoning space. Also, the consequences attached to the misunderstanding are small, as there will most likely be signs indicating floor levels in the building, or other people present who could be asked. This together makes for an ideal first example on which to test the ideas and concepts of this project. It is also extremely relevant: there are many situations of any level of complexity where people misunderstand each other, even when they are speaking the same language, because

[1] Source was last checked on 11/07/2019.

Figure 3.1: *The definition of first floor as provided by the Merriam Webster dictionary.*

there is a different meaning attached to an utterance by cultural use (Hofstede and Hofstede, 2005).

3.1.2 Analysis of the Situation

This situation has two agents present, the American one and the British one. There are two protocols in the situation, the American one and the British one. The American agent knows the American protocol, and the British agent knows the British one.

In the most basic permutation of this situation, neither agent knows what protocol the other follows, but both believe the other to be aware of the same protocol as they themselves have. We could even go so far as to say that the agents are not aware of the possibility that there exist more protocols than the one they themselves are aware of.

Because they both believe that the other agent is following the same protocols as they are, they will not feel the need to share more information to make the shared information less ambiguous. Additionally, they will not be aware that there had been ambiguity until they show up for the meeting and do not see the other agent.

There is one piece of communication between the agents, and no other information present that could produce uncertainty in any way. The communication is an announcement, and the message is guaranteed to arrive.

This example was chosen for being very simple, and for containing only minimal reasoning. In the situation as described above there is little complex reasoning going on. Both agents are aware of a protocol, though of a different one. However, both protocols have the same preconditions. Both agents have comparable assumptions about the other agent's beliefs, and neither agent will initially notice that the other agent has reached a different conclusion.

This can also be seen as unconscious assumption of Grice's maxims for cooperative conversation (Grice, 1975). The maxim for quantity states that the contribution to the conversation should be as informative as required, without providing too much information. Applied to this example, this means that both agents will assume that the statement given ('the meeting will be on the first floor') contains all information necessary to make it to said meeting.

If the possibility of reasoning about the other's beliefs is added, this situation does not change. This is because they are both still operating under the assumption that there is only one protocol applicable in this situation.

However, if their beliefs change, more can be gained from this situation. For example, if an agent a is familiar with the existence of another protocol, even without the knowledge of which

protocol the other agent will follow, agent a may take this possibility into account and provide extra information in addition to the initial announcement. Another option is that the agents do not take it for granted that the other agent follows any protocol, and adjust their beliefs accordingly.

The hidden information is unintentionally hidden: neither agent is aware of the fact that there could be other interpretations of the information provided. Going back to Baltag's types of epistemic changes in Subsection 2.2.3, this is misinformation, also known as information-loss, and is caused by holding inaccurate beliefs.

These beliefs can be corrected if one of the agents provides extra information, or if, at the time of the meeting, the agents follow the signs in the building, or any similar solution. For these solutions to work, they need to be present. Those from outside the influence of the agents (such as signs in a building) can just happen, but those from the other agents will not happen as they are not aware of the necessity for sharing more information.

There are many extra options and beliefs that can be investigated, especially when starting from different belief states, but to keep it simple, we will keep to the initial framing of the situation.

3.2 Social Situation Example

This example illustrates a situation where people *intentionally* do not share protocols with each other, and depend on the fact that sometimes protocols are hidden. This section contains a description of the situation, followed by an analysis of the reasoning in the situation.

3.2.1 The Situation

This situation is taken from Van Ditmarsch et al. (2014), and is also mentioned in the literature section of this thesis. It goes as follows:

Example 5 (The voice of Kathleen Ferrier)**.** *Consider a café in the 1950s with three persons, Kate, Jane, and Anne, sitting across a table. Suppose Kate is gay and wants to know whether either of the other two is gay. She wants to convey the right information to the right person, without the other getting any idea of the information that is being communicated. She states: 'I am musical, I like Kathleen Ferrier's voice'. Jane, who is gay herself, immediately realizes that Kate is gay, whereas, for Anne, the statement just conveys a particular taste in music. (Van Ditmarsch et al., 2014, p. 18-19).*

Kathleen Ferrier was a contralto singer in the 1940's and early '50's. A contralto voice is the lowest natural voice type a woman is generally born with. Partly because of the deep sound of her voice, she was considered a lesbian icon at the time. Referring to her in such a way as in the example above would have been considered a signal for people within the lesbian community at the time.

The way this example is framed in Van Ditmarsch et al. (2014) includes reference to the fact that this café conversation takes place in the Netherlands in the 1950s. They base the context for this example on Van Kooten Niekerk and Wijmer (1985), which discusses the life of lesbians in the Netherlands between the 1920s and the 1960s. During that time there were heavier social and economic consequences attached to being out as lesbian, so there is a higher incentive to make sure nobody finds out than would be the case now. While it was not illegal to be homosexual, there was still heavy discrimination in general society, and in several places one could not be out as gay and keep their job.

Even though it was not illegal to be gay, gay people, especially men, were still frequently arrested and imprisoned. The reason cited would 'public indecency', of which the definition is vague enough that truly innocent gestures could be read as indecent. The age of consent for homosexual couples was also higher than for heterosexual couples (21 instead of 16). This was also frequently used to arrest homosexuals. Police officers would go undercover posing as an interested young homosexual, and would proposition older (mostly) men. Any who responded to that were jailed for breaking the law.

Seen from our 21st century point of view, the secrecy involved in the example mentioned above might seem to be used as a fun way to find out if others are gay. In reality, in the '50s this level of secrecy could on occasion be necessary if one did not want to be arrested.

As briefly mentioned in Section 2.2.4, there is a hidden protocol, complete with motivations and consequences, in this situation. It is slightly more complex than the example explained in Section 3.1. There are three participants in this situation, and the protocol is intentionally hidden. This in contrast to the previous example, where the participants were simply not aware of the possibility that their protocols could differ. There, all agents were lacking information, whereas here some agents have more information and awareness of protocols than the other agents.

3.2.2 Analysis of the Situation

There are three agents in this situation: Kate, Jane and Anne. There is one protocol: "If an agent i states 'I am musical, I like Kathleen Ferrier's voice', then agent i is gay". Kate and Jane are gay. They are both aware of the protocol. Anne is not gay, and is also not aware of the protocol. We assume that all agents know about themselves whether or not they are gay.

Because Anne is unaware of the protocol, she is also unaware of any connection from the conversation she is having with friends and the fact that one or more of them might be gay. As far as Anne is concerned, there is and will be nothing suspicious going on. She is having coffee with some friends and talking about anything and nothing, of which one topic happens to be music.

On the other hand for Kate and Jane, who are both aware of the protocol, there is more involved in the situation. Kate is planning to try to find out if one of the others is gay, and wants to do so without accidentally outing herself to people who are not also gay.

The reasoning in this situation does not start until Kate makes her announcement. All conversation that is irrelevant to application of the protocol is ignored in the epistemic representation of this situation.

After Kate makes her announcement, the following reasoning takes place. Jane observes that the preconditions for the protocol she knows are filled, and therefore reasons that Kate is gay. For Kate and Anne nothing changes.

The interesting part here is that in the situation as described above, there is no acknowledgement coming from Jane that she has understood and received the hidden information. So even though Kate made the statement to find out whether one of the others was gay, she only succeeded in outing herself to those in the know. Ideally, Jane would give some meaningful response that would indicate to Kate that she both understands and is also gay. Looking back at the footnote mentioned in Van Ditmarsch et al. (2014), this could be a message such as 'I agree, I especially liked her performance as Orfeo in Gluck's Orfeo ed Euridice'.

The full example for this situation would then be:

Example 6 (Kathleen Ferrier's voice, portrayal of Orfeo)**.** *Consider a café in the 1950s, with three persons, Kate, Jane and Anne sitting across a table. Suppose Kate is gay and wants to know whether either of the other two is gay. She wants to convey the right information to the right person, without the other getting any idea of the information that is being communicated. She states: 'I am musical, I like Kathleen Ferrier's voice'. Jane, who is gay herself, immediately realizes that Kate is gay, whereas for Anne, the statement just conveys a particular taste in music.*

Jane immediately wants to make sure that Kate knows she is gay, so she responds 'I agree, I especially liked her performance of Orfeo in Gluck's Orfeo ed Euridice'. The role of Orfeo, the male lead, is the one Kathleen Ferrier portrayed, and so to Kate it is clear that Jane is also gay. Meanwhile, Anne is still oblivious to the hidden message communicated.

If an acknowledgement is added and performed, Kate will then know that Jane is also gay, and poor Anne will still be none the wiser.

The discussion above shows that Kate considers the fact that one of the other agents might also be gay. If she did not consider that, she would not have made her statement. It is clear that while she does not know that they are aware of the protocol, she also does not know if they don't know, so she tries out the message in hopes that one of the other agents will respond.

In the above examples, Anne is oblivious. It is, however, interesting to consider what would

happen if Anne happened to be an undercover police officer, trying to infiltrate the lesbian community. Would the protocol still be able to work?

If Kate and Jane have no suspicions about Anne, then they would likely perform the same actions. It is unclear how much information such a police officer would need before being able to arrest them for public indecency. Even if the officer is aware of the protocol, she cannot use it to prove anything, as it is a seemingly normal conversation about music.

Of course, being arrested on the spot is not the only concern of Jane and Kate, they also do not want to be socially ostracised, or fired from their job. The best they can possibly do is to try not to get infiltrated, or be very cautious about the situations in which they use the protocols. As this one is, as mentioned, 'just' a reference to a music preference, it was probably one of the safer ones at the time.

Whichever way they would attempt to solve this, the reasoning in the situation would become quite a lot more complex, because it will involve reasoning with suspicions, aside from just reasoning about secret information. This may result in trying to communicate secretly, and definitely involves reasoning about the motives of the other agents.

3.3 Diplomatic Situation

The idea behind this third example was to have an example that models a diplomatic situation. Another possible way of describing this is as an example in which the key point of the example is negotiation. It is an example wherein all parties have incentive to cooperate partially, while still retaining part of a competitive stance. Compared to previous examples, the agents are neither hiding all information, nor are they open about all information: it is a balancing game.

3.3.1 The Situation

Instead of coming up with a concrete diplomatic situation that needs to be negotiated properly, either from a real world or a fictional example, this example will use a negotiation game as analogy. The game is known as coloured trails (De Weerd et al., 2017). It is designed to model a prototypical multi-issue bargaining situation. The basic situation is a rectangular board made up of tiles of different colours. Agents playing all have a goal and a starting point, and they can reach their goal by taking steps across the coloured tiles. To cross a tile, the agents have to hand a coloured chip in the same colour as the tile they want to step on to. There is an initial distribution of chips, but agents can barter chips with each other in order to get closer to their goal. The game is designed to be adaptable to many different types of bargaining situations, so we can tweak it to be most useful to the situation at hand.

The reason it is preferable over finding a 'real' example to apply it to is that due to the set-up of the game, there is a scoring mechanism. This scoring system allows us to reason about a mixed-motive situation and be able to reach a solution, due to the inherent prioritisation that is necessary to resolve situations like that.

In this game, there is a 5 x 5 board of coloured tiles. Each of the two agents playing will receive a goal to attempt to reach, which will be at least three steps away from the starting point. The agents will not know the other agent's goal. Each agent will start in the centre of the board, and is given four chips coloured in the same colours as the tiles of the board. The agents need to use the chips to get as close to the goal as possible, and they can only travel over tiles that are adjacent to their current tile and that they have a matching chip for. They need to hand in a chip to pass that tile. The agents receive points for getting closer to their goal, for reaching their goal, and for retaining unused chips at the end of the game. The points awarded are so distributed that the highest incentive is to reach the goal, but that there is still incentive to have unused chips.

An agent will receive 100 points for every step they take towards the goal. They will get 500 on top of that if they reach the goal. Finally, they get an extra 50 points for every chip they still have in their possession after handing in the ones needed to travel across the board. To be able to reach their goal, they can swap chips with the other agent. This proceeds in a series of bargaining offers. An agent starts with an offer. The other agent can decide whether they want to accept this offer or not. For this they have three options. An agent can choose to accept the offer, they can reject the offer and close negotiations, or they can reject the offer and do an offer of their own.

The bargaining offers are based on the chips an agent has, the chips the other agent has, and the goal an agent has. Because an agent does not know what goal the other agent is aiming towards, that is less easy to take into account. There are no restrictions pertaining to the content of an offer. An agent can repeat their own offer, or even suggest an offer they themselves have rejected earlier. The only caveat is that there is a point deduction of 1 for each agent per round of negotiation. This is marginal compared to the points that can be received in-game, and mostly

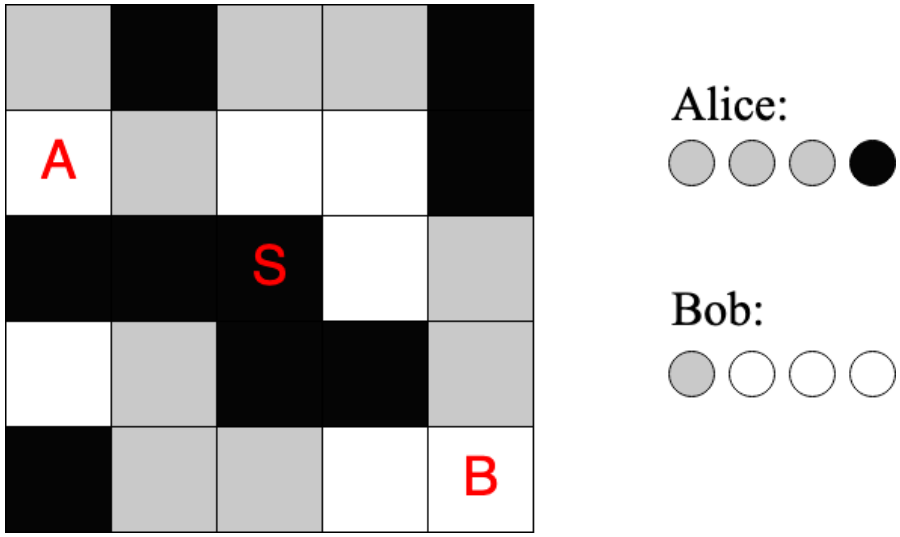

Figure 3.2: *An example of the coloured trails game. This image is adapted from De Weerd et al. (2017).*

serves to make infinite negotiations unappealing, and to promote cooperation.

The specific example we chose to use is the one shown in De Weerd et al. (2017), and can be seen here in Figure 3.2. It has three colours: white, grey, and black. Consequently, the chips that the agents have will also be in these colours. There are two agents, Alice and Bob. Alice's goal is shown by the red **A** in the current image, Bob's goal is in the lowermost right hand corner, shown by a red **B**. For both agents the starting point is in the middle, denoted by the red **S**. This also means that to reach their goals, Alice has three steps to make, and Bob has four steps to make. The chip distribution is visible to all agents, but the goals are only visible to the agent that has that goal. The chips in the game allow for both agents to reach their goal, if they reach an optimal distribution of chips. This leaves one chip extra that can give an agent extra points. In the paper this was taken from, the authors do not mention the exact goal of Bob. However, as we need a specific goal to best model the reasoning, we chose a goal for Bob.

Using three colours and four chips as in the example provided makes reaching the goal plausible with the number of chips in play. Making the chips visible to all agents increases the potential for optimal negotiation, and hiding the goals is in line with the hidden information this thesis is discussing. We decided to have the chips open to improve the quality of negotiation in this example, and to make it easier to reason about. The total possible search space will be smaller, and therefore more manageable.

This coloured trails example can be seen as an analogy for a real-world example. The coloured trails game is used as analogy for task distribution and negotiation, among others (Gal et al., 2010). For a specific example, take the following example[2].

Example 7 (Negotiation over a Quarry and a Forest)**.** . *Imagine an earl and a prior in the middle ages. The earl is the owner of a large piece of ground, including a town and a quarry. In that town is a priory, which owns a small forest. The priory wishes to build a new chapel as*

[2]This example is adapted and much simplified from the historical fiction novel 'Pillars of the Earth', by Ken Follett.

the previous one had been destroyed in a fire, and the prior wishes the new chapel to be worked from stone, so it is less likely to burn down. He does not have the money to pay the earl to mine the stone from the quarry, so wishes to negotiate a different deal. Meanwhile, the earl wants to continue owning the quarry as it is an easy way to make a profit. However, he wants to own the forest so he can hunt in it. The prior does not want to sell the forest to the earl, because he needs the wood from the forest to build scaffolding for the chapel. After harsh back and forth negotiation, they reach a compromise: the quarry remains in possession of the earl, but the priory is allowed to mine stone there without having to pay. In return, the earl and his hunting companions are allowed to hunt in the forest without punishment.

This negotiation involves a lot of different elements: there is the possessions both agents have, and the goals they have. Where at first glance one could conclude that both the prior and the earl want to own both the quarry and the forest, this is not exactly true. As that would not have led to a productive result, they had to negotiate about what exactly they wanted to accomplish. In the coloured trails game, the chips the agents had would represent the different aspects of what they owned. So for the prior, owning the forest meant being able to hunt in the forest, and to take wood from the forest. For the earl, owning the quarry would mean being able to mine stone himself, and being able to charge others for mining stone there. When these chips get exchanged, both agents can reach their goals.

3.3.2 Analysis of the Situation

There are two agents, Alice and Bob. Both agents can see the board, and the chips that they both possess. They know their own goal, but cannot see what goal the other is aiming for. They know the rules of the game (and know that the other knows the rules of the game), including the point system.

The protocols in this example are a lot more complicated than in the previous examples. The protocols here need to enable the agent to play the game, to do the bargaining towards an ideal situation. To model a game like this in protocols, we use different types of protocols. First, there are the more general behavioural protocols, relating the rules of the game to preferences in bargaining. The other protocols are specific per agent, and deal with the different options and preferences they have for reaching the specific goal they are aiming for.

These protocols together should model the playing situation in a manner that is similar to how the game would normally be played. These protocols will be more completely defined in Section 5.3.

In the simplest case – the one that we will formalise first – agents are not able to reason about the protocol beliefs of the other agents. In this case, this also means they cannot reason about the goal of the other agent, or that the other agent might even be aiming towards a goal. When looking back at the De Weerd et al. (2017) paper, this is surprisingly similar to the case for agents that possess no Theory of Mind.

If we add the beliefs on protocols to this situation, we are allowing the agents to reason about other agents' belief and knowledge. In effect, we are creating Theory of Mind. We could add just one layer: 'I believe the other agent has goal G', or two: 'The other agent can deduce the following information about my goals, given my actions'.

To translate this into a protocol framework, we would replace sentences like 'I believe Bob is aiming for goal G' with: 'I believe Bob follows protocol G_1', where G_1 is the protocol that assigns high priority to reaching G from the starting point. Because the translation results in reasoning about protocols, it is a better fit with the set up of the reasoning in this thesis. This information can then be used to create bargaining offers that they believe the other agent is more

likely to accept, based on beliefs about their goals.

It is possible to add the second layer as well, and its use is a lot more complex. The beliefs about what other agents can deduce about your own goals can be used in different ways, depending on the mindset one starts with. So if an agent is cooperatively oriented, (and believes the other is so too), they might choose to make offers that will give the other agent as much information as possible, so as to better (and earlier) reach a profitable conclusion.

However, if the agents are suspicious of each other, as they could be in a real diplomatic negotiation, it would be different. Instead of providing as much information as possible, they would choose to minimise information sharing. This might hinder them in reaching their goal in a timely (or efficient) manner. On the other hand, if an agent is maliciously oriented, they can really use beliefs on what goal the other agent is aiming for to give them chips that will not get them far.

If one agent is cooperative, and the other suspicious and/or malicious, the results would be interesting to watch.

Chapter 4

Theoretical Framework

In this chapter, we will give a description of the logic system designed by Van Ditmarsch et al. (2014), Epistemic Protocol Logic. In the next chapter, this system will be used to create a formalisation of the examples. The formal definitions in this section are adapted from Van Ditmarsch et al. (2014) with permission from the authors. We use the formalisation here, while full explanations and proofs for this system can be found in Van Ditmarsch et al. (2014).

4.1 Models and Observation

In this section we will introduce epistemic expectation models and observation. Epistemic expectation models are Kripke models that are capable of modelling expected observations. Observations are, in this case, the information agents perceive from their environment.

To begin with, let $\mathbf{I}$ and $\mathbf{P}$ respectively be a finite set of agents and a finite set of atomic propositions. These propositions describe the facts of the world. Let $Bool(\mathbf{P})$ denote the set of all Boolean formulas over $\mathbf{P}$.

Definition 2 (Epistemic Model)**.** An epistemic model $\mathcal{M}_e$ is a triple $\langle S, \sim, V \rangle$, with S a non-empty domain of states, $\sim$ a set of accessibility relations $\{\sim_i \mid i \in \mathbf{I}\}$, and $V : S \to \mathcal{P}(\mathbf{P})$ a valuation assigning to each state a set of propositional variables that are true in that state.

The definition we are working towards here specifically is that of the epistemic expectation models. These show the expectations the agents have of the world based on the observations they make. To properly do this, we need to define what an observation and observation expression is and can be. Observations can be of facts or of actions. Observations of facts are information such as 'The window is open', whereas action observations are more like 'making an announcement' or 'winking at an agent'. For formalisation purposes there is no difference between the types. We will focus mostly on the action-related observations. These are based on the (finite) set of possible actions within the environment. These observations can be observed sequentially and combined as follows.

Definition 3 (Observation expressions)**.** Given a finite set of atomic action symbols $\mathbf{\Sigma}$, the language $\mathcal{L}_{obs}$ of observation expressions is defined as follows:

$$\pi ::= \delta \mid \varepsilon \mid a \mid \pi \cdot \pi \mid \pi + \pi \mid \pi^*$$

with δ as the empty set $\emptyset$ of observations, ε a constant representing the empty string, and $a \in \mathbf{\Sigma}$. Both δ and ε are used, because in a later stage there will be a reduction over observations, and

we will need to be able to differentiate between an empty set of observations and the empty string of an observation.

For this definition of observation expression we need to define the semantics. This can be done by describing the semantics for sets of observations.

Definition 4 (Semantics of Observations)**.** The set of observations $\mathcal{L}(\pi)$ connected to observation expression π is a set of finite strings over $\mathbf{\Sigma}$, defined inductively as:

$$\begin{aligned}
\mathcal{L}(\delta) &= \emptyset \\
\mathcal{L}(\varepsilon) &= \{\varepsilon\} \\
\mathcal{L}(a) &= \{a\} \\
\mathcal{L}(\pi \cdot \pi) &= \{w\, v \mid w \in \mathcal{L}(\pi) \text{ and } v \in \mathcal{L}(\pi')\} \\
\mathcal{L}(\pi + \pi) &= \mathcal{L}(\pi) \cup \mathcal{L}(\pi')\} \\
\mathcal{L}(\pi^*) &= \{\epsilon\} \cup \bigcup_{n>0} (\mathcal{L}(\underbrace{\pi \ldots \pi}_{n}))
\end{aligned}$$

Next, we can use the concept of observations, combined with epistemic models, to create epistemic expectation models.

Definition 5 (Epistemic Expectation Model)**.** An epistemic expectation model $\mathcal{M}_{exp}$ is constructed from an epistemic model and an expected observation function to form the quadruple $\langle S, \sim, V, Exp\rangle$. From this, $\langle S, \sim, V\rangle$ is the epistemic model, and Exp is the expected observation function. $Exp : S \to \mathcal{L}_{obs}$ assigns an observation expression π to each state such that $\mathcal{L}(\pi) \neq \emptyset$. Formally, Exp goes to one expected observation. Intuitively, it can be read as going to a set of potential expected observations. Furthermore, an epistemic expectation state is a pointed epistemic expectation model $\langle S, \sim, V, Exp, s\rangle$, for state s.

An epistemic expectation model can sometimes also be written as $(\mathcal{M}_e, Exp)$, where $\mathcal{M}_e$ stands for the epistemic model. Furthermore, $\mathcal{M}_e$ can be seen as an $\mathcal{M}_{exp}$ in which for all $s \in S$, $Exp(s) = \mathbf{\Sigma}^*$. In this, $\mathbf{\Sigma}^*$ denotes $(a_0 + a_1 + \cdots + a_k)^*$, where $\mathbf{\Sigma} = \{a_0, \ldots, a_k\}$.

As this logic is eventually going to be used in a changing system, it is necessary to have a definition of how the model will be updated. The only changes possible thus far are through the observations. In Section 2.3.6 we explained how agents remove inaccessible words after an announcement is made. The updates for observations work in a similar fashion. After observing actions, agents remove the scenarios that have become impossible in their epistemic expectation model. Those 'impossible' states are states in which the noticed observation could not have happened. The removal of states in this way can be formalised as in the following definition.

Definition 6 (Update by Observation)**.** Let $\mathcal{M} = (S, \sim, V, Exp)$ be an epistemic expectation model, and take $w \in \mathbf{\Sigma}^*$ to be an observation over $\mathbf{\Sigma}$. When the model is updated, it becomes: $\mathcal{M}|_w = (S', \sim', V', Exp')$, with $S' = \{s \mid \mathcal{L}(Exp(s)\backslash w) \neq \emptyset\}$, $\sim_i' = \sim_i |_{S' \times \mathbf{I} \times \mathbf{S}'}$, $V' = V|_{S'}$, and $Exp'(s) = Exp(s)\backslash w$. In this, $\pi\backslash w$ is defined as the set $\{v \mid w\, v \in \mathcal{L}(\pi)\}$.

This is done with the help of output function o that maps regular expressions π to ε or δ, according to the following definition. In all cases these rules behave as reduction algorithms on the $\backslash a$ operation, pushing it further inward, so as to be able to eliminate them. This allows us to compute the residuals of observations on a syntactical level.

$$\pi = o(\pi) + \sum_{a \in \mathbf{\Sigma}} (a \cdot \pi \backslash a) \qquad \varepsilon \backslash a = \delta \backslash a = b \backslash a = \delta \ \ (a \neq b)$$

$$o(\varepsilon) = \varepsilon \qquad a \backslash a = \varepsilon$$

$$o(\delta) = o(a) = \delta \qquad (\pi + \pi') \backslash a = \pi \backslash a + \pi' \backslash a$$

$$o(\pi + \pi') = o(\pi) + o(\pi') \qquad (\pi \cdot \pi') \backslash a = (\pi \backslash a) \cdot \pi' + o(\pi) \cdot (\pi' \backslash a)$$

$$o(\pi \cdot \pi) = o(\pi) \cdot (\pi') \qquad \pi^* \backslash a = \pi \backslash a \cdot \pi^*$$

$$o(\pi^*) = \varepsilon \qquad \pi \backslash a_0 \cdots a_n = \pi \backslash a_0 \backslash a_1 \ldots \backslash a_n$$

Now, we define the following Public Observation Logic (POL) to reason about the observations.

Definition 7 (Public Observation Logic)**.** The formulas φ of POL can be given by:

$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid [\pi]\varphi$

with $p \in \mathbf{P}$, $i \in \mathbf{I}$, and $\pi \in \mathcal{L}_{obs}$. We use the usual abbreviations. $K_i\varphi$ means that agent i knows φ, and $[\pi]\varphi$ stands for 'after announcement of π, φ holds'.

From this, we also get a truth definition for POL.

Definition 8 (Semantics of POL)**.** Given an epistemic expectation model $\mathcal{M} = (S, \sim, V, Exp)$, a state $s \in S$, and a POL-formula φ, the definition of $\mathcal{M}, s \vDash \varphi$ is as follows:

$$\begin{array}{lll}
\mathcal{M}, s \vDash p & \text{iff} & p \in V(s); \\
\mathcal{M}, s \vDash \neg\varphi & \text{iff} & \mathcal{M}, s \nvDash \varphi; \\
\mathcal{M}, s \vDash \varphi \wedge \psi & \text{iff} & \mathcal{M}, s \vDash \varphi \text{ and } \mathcal{M}, s \vDash \psi; \\
\mathcal{M}, s \vDash K_i\varphi & \text{iff} & \text{for all } t \in S : (s \sim_i t \text{ implies } \mathcal{M}, t \vDash \varphi); \\
\mathcal{M}, s \vDash [\pi]\varphi & \text{iff} & \text{for each } w :\in \mathcal{L}(\pi) : (w \in init(Exp(s)) \text{ implies } \mathcal{M}|_w, s \vDash \varphi),
\end{array}$$

where $w \in init(\pi)$ iff $\exists v \in \mathbf{\Sigma}^*$ such that $wv \in \mathcal{L}(\pi)$.

4.2 Protocols and Expressions

The observations defined in the above subsection can give agents information about the current environment. The way agents use these observations is dependent on the protocols used. These protocols are instrumental in allowing agents to understand the implications of their observations. Furthermore, the protocols can also stimulate actions and responses, which in turn provide observational material for the other agents.

A protocol is a rule, defining the consequences or required responses for specific conditions. To define protocols properly, we specify them in a language of protocol expressions $\mathcal{L}_{prot}$.

Definition 9 (Protocol Expression)**.** The language $\mathcal{L}_{prot}$ can be defined as follows:

$\eta ::= \delta \mid \varepsilon \mid a \mid ?\varphi \mid \eta \cdot \eta \mid \eta + \eta \mid \eta^*$

with δ for empty language $\emptyset$, ε as a constant representing an empty string, and $\varphi \in Bool(\mathbf{P})$.

The protocol language is very similar to the one for observation expressions, though for protocols instead of observations. Additionally, a Boolean test is added. These tests allow for conditions to be set, so we can create statements such as 'if it is raining, we will take an umbrella, and if it is not raining, we do not take an umbrella': $(?rain \cdot umbrella) + (?\neg rain \cdot \neg umbrella)$.

Going back to protocols, the tests help describe the pre-conditions for observations to happen. If a protocol has no tests, that means there are no requirements to be met before the observation can happen.

To determine how the expected observations connect to a protocol, we need the semantics for protocol expressions. A protocol η can be seen as a set $\mathcal{L}_g(\eta)$ of guarded observations in the form:

$$\rho_0 a_0 \rho_1 a_1 \dots \rho_k a_k$$

in which case each ρ represents the pre-conditions necessary for the action to occur (and therefore the connected observation to happen). All $\rho_i \subseteq \mathbf{P}$ connect to their matching a_i, and can be read as that all propositions $p \in \rho_i$ need to be true for a_i to occur. For Boolean formulas φ, it works similarly. In this case: if $\rho \vDash \varphi$ following Definition 8.

Definition 10 (Semantics of Protocol Expressions)**.** The set of guarded observations $\mathcal{L}_g$ corresponding to protocol expressions are defined as:

$$\begin{aligned}
\mathcal{L}_g(\delta) &= \emptyset, \\
\mathcal{L}_g(\varepsilon) &= \{\rho \mid \rho \subseteq \mathbf{P}\}, \\
\mathcal{L}_g(a) &= \{\rho a \rho \mid \rho \subseteq \mathbf{P}\}, \\
\mathcal{L}_g(?\psi) &= \{\rho \mid \rho \vDash \psi,\ \rho \subseteq \mathbf{P}\}, \\
\mathcal{L}_g(\eta_1 \cdot \eta_2) &= \{w \diamond v \mid w \in \mathcal{L}_g(\eta_1) \text{ and } v \in \mathcal{L}_g(\eta_2)\}, \\
\mathcal{L}_g(\eta_1 + \eta_2) &= \mathcal{L}_g(\eta_1) \cup \mathcal{L}_g(\eta_2)\} \\
\mathcal{L}_g(\eta^*) &= \{\rho \mid \rho \subseteq \mathbf{P}\} \cup \bigcup_{\mathbf{n>0}} (\mathcal{L}_{\mathbf{g}}(\eta^\eta))
\end{aligned}$$

where $\diamond$ is the fusion product: $w \diamond v = w'\rho v'$ when $w = w'\rho$ and $v = \rho v'$, and not defined otherwise.

Using this we can derive the expected observations according to η based on condition ρ using a conversion function f_ρ. We will need this conversion function when performing updates on the models.

Definition 11 (Conversion Function)**.** The conversion function $f_\rho : \mathcal{L}_{prot} \to \mathcal{L}_o bs$ is defined as follows:

$$\begin{aligned}
&f_\rho(\delta) = \delta && f_\rho(\eta \cdot \eta') = f_\rho(\eta) \cdot f_\rho(\eta') \\
&f_\rho(\varepsilon) = \varepsilon && f_\rho(\eta + \eta') = f_\rho(\eta) + f_\rho(\eta') \\
&f_\rho(a) = a && f_\rho(\eta^*) = (f_\rho(\eta))^* \\
&f_\rho(?\varphi) = \begin{cases} \varepsilon & \text{if } \rho \vDash \varphi \\ \delta & \text{else (i.e. if } \rho \nvDash \varphi) \end{cases}
\end{aligned}$$

The next step is to introduce epistemic protocol models. These will allow us to reason about uncertainty about protocols.

Definition 12 (Epistemic Protocol Model)**.** An epistemic protocol model $\mathcal{A}$ is a triple $\langle T, \sim, Prot \rangle$. In this, T is a domain of abstract objects, $\sim$ is a set of accessibility relations $\{\sim_i \mid i \in \mathbf{I}\}$, and $Prot : T \to \mathcal{L}_{prot}$ assigns a protocol to each domain object.

These epistemic protocol models can show the uncertainty between protocols, but that does not yet map directly onto our epistemic expectation models. For that, there is a protocol update. This defines how an epistemic protocol model can be applied to an epistemic expectation model and combine to create a new epistemic expectation model with the effects of the protocol updated.

Definition 13 (Protocol Update)**.** Taking an epistemic protocol model $\mathcal{A} = \langle T, \sim, Prot \rangle$ and an epistemic expectation model $\mathcal{M}_{exp} = \langle S, \sim, V, Exp \rangle$, the product $(\mathcal{M}_{exp} \otimes \mathcal{A} = (S', \sim', V', Exp')$ is defined with:

$$S' = \{(s,t) \in S \times T : \mathcal{L}(f_{V_{\mathcal{M}}(s)}(Prot(t))) \neq \emptyset\};$$
$$(s,t) \sim_i' (s',t') \text{ iff } s \sim_i s' \text{ in } \mathcal{M}_{exp} \text{ and } t \sim_i t' in \mathcal{A};$$
$$V'(s,t) = V(s);$$
$$Exp'((s,t)) = f_{V_{\mathcal{M}}(s)}(Prot(t)).$$

Continuing on from Public Observation Logic, we can now define Epistemic Protocol Logic (EPL).

Definition 14 (Epistemic Protocol Logic)**.** The formulas φ of EPL can be given by:

$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_i\varphi \mid [\pi]\varphi \mid [!\mathcal{A}_e]\varphi$

with $p \in \mathbf{P}$, $i \in \mathbf{I}$, and $\pi \in \mathcal{L}_{obs}$. $\mathcal{A}_e$ is an epistemic protocol with designated state e.

Next, there is a slight addition to the semantics of POL to create a full truth definition for EPL.

Definition 15 (Semantics of EPL)**.** Given an epistemic expectation model $\mathcal{M} = (S, \sim, V, Exp)$, a state $s \in S$, and a EPL-formula φ, the definition of $\mathcal{M}, s \vDash \varphi$ is as defined for POL for the most part (see Definition 8). For the formulas they have in common, nothing changes. That only means a truth definition for $[!\mathcal{A}_e]\varphi$ is added:

$$\mathcal{M}, s \vDash [!\mathcal{A}_e]\varphi \text{ iff } \quad \text{If } \mathcal{L}(f_{V(s)}(Prot(e))) \neq \emptyset \text{ then } \mathcal{M} \otimes \mathcal{A}, (s,e) \vDash \varphi$$

Colloquially, this means that $[!\mathcal{A}_e]\varphi$ can be read as: after installing new epistemic protocol $\mathcal{A}_e$, φ holds. Installing of protocols in this way is done because we want to be able to show the protocols that exist, or that agents have. This also makes it possible to change a situation by adding new or more protocols.

4.3 Factual Changes

Up until this point, all changes in the system have been epistemic. In some situations, however, there can be factual changes. These changes are changes that affect the real world. An example is a window being opened. Initially the window is closed, and when it is opened, the situation has changed. In the system as described in Van Ditmarsch et al. (2014), agents can only observe actions, and cannot observe the world itself. This means that to know whether the window is open or closed can be determined by the action observed. We will keep to that description.

The first element to introduce is a definition of fact-changing actions.

Definition 16 (Fact-Changing Actions)**.** A set of fact-changing actions is a tuple $(\mathbf{\Sigma}, \iota)$ such that $\iota : \mathbf{\Sigma} \times \mathbf{P} \to \mathbf{Bool}(\mathbf{P})$.

This can be read as that ι shows the post-conditions of actions. After performing action a, the new truth distribution is defined by $\iota(a, p)$. This is restricted to just Boolean values. Actions

that have only purely epistemic effects can be modelled as follows: $(\mathbf{\Sigma}, \iota_{\mathbf{0}})$, in which for each $a \in \mathbf{\Sigma}$, $\iota_0(a)$ is the identity function.

For ease of use in the later update model, we will introduce a factual change system. It holds the information contained in fact-changing actions.

Definition 17 (Factual Change System)**.** A factual change system $\mathcal{F}$ on $\mathbf{\Sigma}$ is a tuple (Q, r) with $Q = \mathcal{P}(\mathbf{P})$ and function $r : Q \times \mathbf{\Sigma} \to Q$.

In effect, a factual change system contains the factual change information on all propositions, both those contained by ρ and those not affected.

From this and Definition 13, we now come to the final definition: how to update a model in a protocol-situation, taking into account the possibility of factual changes. In this, we will use $\mathcal{F}$ and $\mathcal{G}$ in superscript with a model to show it is a model that also has a factual change system.

Definition 18 (Protocol Update with Factual Changes)**.** Taking a fact-changing epistemic expectation model $\mathcal{M}^{\mathcal{F}}_{exp} = \langle S, \sim, V, Exp, \mathcal{F}\rangle$, and a fact-changing epistemic protocol model $\mathcal{A}^{\mathcal{G}} = \langle T, \sim, Prot, \mathcal{G}\rangle$, the product $(\mathcal{M}^{\mathcal{F}}_{exp} \otimes \mathcal{A}^{\mathcal{G}}) = (S', \sim', V', Exp', \mathcal{F}')$ is defined as follows:

$$
\begin{array}{ll}
S' & = \{(s,t) \in S \times T : \ \mathcal{L}(f_{V_{\mathcal{M}}(s)}(Prot^{\mathcal{G}}(t))) \neq \emptyset\}; \\
(s,t) \sim'_i (s',t') & \text{iff } s \sim_i s' \text{ in } \mathcal{M}_{exp} \text{ and } t \sim_i t' \text{ in } \mathcal{A}; \\
V'(s,t) & = V(s); \\
Exp'((s,t)) & = f_{V_{\mathcal{M}}(s)}(Prot^{\mathcal{G}}(t)); \\
\mathcal{F}' & = \mathcal{G},
\end{array}
$$

with $Prot^{\mathcal{G}}(t)$ is the normal form of $Prot(t)$ with respect to $\mathcal{G}$.

These additions change some of the semantics for EPL. Those changes are as follows.

Definition 19 (Semantics of EPL with factual changes)**.**

$$
\begin{array}{lll}
\mathcal{M}^{\mathcal{F}}_{exp}, s \vDash [!\mathcal{A}^{\mathcal{G}}_{e}]\varphi & \text{iff} & \text{If } \mathcal{L}(f_{V(s)}(Prot^{\mathcal{G}}(e))) \neq \emptyset \text{ then } \mathcal{M}^{\mathcal{F}}_{exp} \otimes \mathcal{A}^{\mathcal{G}}, (s,e) \vDash \varphi \\
\mathcal{M}^{\mathcal{F}}_{exp}, s \vDash [\pi]\varphi & \text{iff} & \text{for each } w :\in \mathcal{L}(\pi) : (w \in init(Exp(s)) \text{ implies } \mathcal{M}^{\mathcal{F}}_{exp}|_w, s \vDash \varphi)
\end{array}
$$

with $\mathcal{M}^{\mathcal{F}}_{exp}|_w = (S', \sim', V', Exp', \mathcal{F})$, with $S', \sim', Exp'$ defined as in Definition 6 and $V'(s) = r(V(s), w)$ with r transition function in $\mathcal{F}$.

Now all essential definitions have been defined, it is possible to start to express the example situations in this formalisation.

Chapter 5

Formalisation

For each of the examples we will create a formalisation here. Together with each formalisation we will relate the aspects of the examples (as related in Chapter 3) to what could be formulated. The conclusion to this chapter will be that there is room for improvement in the reasoning that can be expressed in the formalisation in its current state.

In all formalisations, the protocols as defined will be known and acted upon by an overarching system. This system will distribute knowledge to the agents, according to the protocols with satisfied preconditions. The agents will still perform their own actions. This is partially to be able to express the consequences of actions in a better way, and partially so it can be made clear that agents will not gain access to information they should not be gaining. This way, there is no way agents can see into other protocols that are defined. This only increases the strength of the link between the formalisation or simulation and the real world. It also allows us the reasoning space necessary.

In addition to the factual changes described in Definitions 16 and 17, we will be using the epistemic expectation model and epistemic protocol model as defined in Definitions 5 and 12.

5.1 Language Example Formalisation

This model refers back to the language example as described in Section 3.1. The agents have received proper names for ease in referring to them.

$$\begin{aligned} \mathbf{I} &= \{\text{Abe}, \text{Britt}\} \\ \mathbf{P} &= \{c_i,\ n_i,\ g_i\} \\ \mathbf{\Sigma} &= \{f\} \\ V &: \{c_{\text{Abe}} = a,\ c_{\text{Britt}} = b\} \end{aligned}$$

The actions and propositions and their meanings are as follows. The (fact-changing) action in the system is f: the announcement of the fact that the meeting is on the first floor. This action activates the protocol by satisfying its preconditions. The other propositions are n and g. In this, n stands for 'the agent will go to the next floor up from ground level', and g is defined as 'the agent will go to the floor on ground level'. Additionally n_i and g_i are mutually exclusive and exhaustive: an agent i will go to either the ground floor or the next one up, not both and also not neither.

The proposition c_i has to do with the nationality of the agents. Its possible valuations are a and b, e.g. $c_i = a$: the country of agent i is America. Each agent knows their own nationality. In

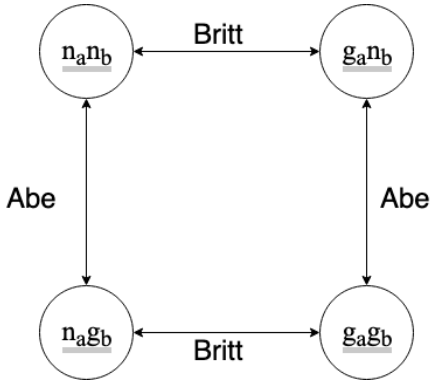

Figure 5.1: *The states for the language example as defined in Section 5.1.*

effect, this is a binary definition of c, as it has only two options, and all agents in this situation have to be one or the other. The states for the agents at the beginning of the example can be seen in Figure 5.1.

To ensure that the agents remain unaware of the differences in protocol known, the system controls the protocol. The protocol is defined as follows:

$$\eta = ?f \cdot \theta_i, \text{ where:}$$
$$\theta_i := ?(K_i\ (c_i = a)) \cdot g_i + ?(K_i\ (c_i = b)) \cdot n_i$$

This is performed for all agents $i \in \mathbf{I}$.

The protocol states that if f is true (is announced), the rest of the protocol will run. This is that if i knows that its country is a (America), then i will go to the ground floor. Also, if i knows that its country is b (Britain), then i will go to the floor above the ground floor.

To create this protocol, a knowledge operator needed to be added to the existing formalisation. This is a very natural extension given the way this formalisation was created initially. Because the protocol framework was created based on principles from Dynamic Epistemic Logic, adding epistemic operators is not too large of a deviation from the original set-up. Additionally, this addition fits intuitively to the way the example situation has been described. Furthermore, to be able to test for knowledge, the test operator $?$ needs to be extended to cover not just boolean propositions.

Postconditions are more complex to define, as there are no absolute truths in this example. The way the example is currently framed does not provide clarity on which of the propositions is 'true'. This example is intended to be ambiguous. The only change brought about by an action is that f activates the protocol. We could then write postconditions as the results of protocol activation.

Because the protocol has two elements, the postconditions are conditional on the nationality of the agents, so there will be two different postconditional statements, dependent on the state of the agent. The complex issue here is that there is no objective truth about what the first floor is. Because of the current definition of n and g, Boolean values can be assigned (conditional on the agents). So n_{Britt} is true after announcement of f. This is because f activates η, and η in turn changes the values of n and g.

The announcement of meeting is inserted into the situation 'from above', it is not one of the agents that adds information. An objective truth could of course be determined if the entity inserting the information is defined to be either American or British. Because the agents' knowledge or belief is based on the protocol information they are made aware of, they can be wrong in their beliefs. Because of this, we use the belief operator. Formally (and improvising on the two-active-protocols-notation):

$$\mathcal{M}, s \vDash [!\eta][f](g_{\text{Abe}} \wedge n_{\text{Britt}})$$

Colloquially, the above statement says that in this model and state it is the case that, taking activation of η, and announcement of f, Abe will go to the ground floor for the meeting, and Britt will go to the next floor up from the ground floor.

If we want to reason about the agents' belief of the protocols other agents may be aware of, we would have to change the current description enough to have all knowledge and belief explicit. This would result in adding more possible worlds and relations to the current model. Currently, there is no reasoning going on, so we do not really need the separate options in our description.

The states can be described as following. Agents cannot distinguish between states in which they have the same value of n or g respectively. Furthermore, they are not even aware of the existence of other possibilities than the ones they currently hold.

5.2 Social Situation Formalisation

As in the first example, a system will be in charge of actually implementing the effects of the protocol. Whereas in the first example, the agents were unaware of the use of a protocol, in the second example the goal of some of the agents is to not let information slip. The exact situation is Example 6, which contains the initial transfer of information as well as the acknowledgement.

$$\begin{aligned}
\mathbf{I} &= \{k, j, a\} \\
\mathbf{P} &= \{\texttt{gay}_i\} \\
\mathbf{\Sigma} &= \{\texttt{musical}, \texttt{orfeo}\} \\
V &: \{\texttt{gay}_k = \text{true}, \texttt{gay}_j = \text{true}, \texttt{gay}_a = \text{false}\}
\end{aligned}$$

The agents are k, j, and a, after the names Kate, Jane, and Anne in the textual version of this example. The only relevant proposition present is whether or not agents are gay, written as $\texttt{gay}_i$. Two actions are possible in this situation. The first, `musical` is a statement: 'I am musical, I like Kathleen Ferrier's voice'. When modelled, it will be noted with a subscript referring to the agent making the statement. Similarly, the statement `orfeo`: 'I agree, I especially liked her performance as Orfeo in Gluck's Orfeo ed Euridice' is used in the same way.

The protocol consists of two separate elements, one for each action. The actions can only occur in sequence, and not simultaneously. Both statements can only be made by agents that are gay. When $[\texttt{gay}]_i$ is announced by agent i, the response from the other agents is dictated by whether or not they are gay. If a gay agent hears the statement, they will know that the speaker of that statement is gay. When agents that are not gay hear it, they will only register the statement as an expression of musical taste, and not the hidden message behind it. This can be formally written down as:

$$[\texttt{musical}]_x : \texttt{?gay}_i \cdot K_i\ \texttt{gay}_x$$

As a response to this, a gay agent can make statement `orfeo`. Again the consequences of this response depend on the gayness of the agent hearing it. If an i agent is gay and has just made the musical statement, i will know both that the other agent is gay, and that the other agent knows that agent i is also gay. This last bit is because this statement is used only as acknowledgement of the `musical` statement

$$[\texttt{orfeo}]_x : \texttt{?gay}_i \cdot (K_i\ \texttt{gay}_x \wedge K_i\ K_x\ \texttt{gay}_i)$$

Because of the conditional connection between the two statements, we need to keep that aspect in the protocol. The `orfeo` statement does not have the same meaning when not used as acknowledgement. There will be one protocol that contains both elements of this protocol, with a temporal operator. This allows the protocol to be installed in one go, which makes the modelling easier.

To capture the conditionality, a temporal operator is used. The one we are using is the past version of operator F. Recall that $F\varphi$ means that in some future, φ holds. The past version, F^{-1} means that in some past, φ was true. Translating this to our example will allow us to say that if `orfeo` is announced at some point after `musical` has been announced, then the postconditions of `orfeo` will hold.

The combined protocol would look something like this:

$$\begin{aligned}\eta &= \mathbf{?}[\texttt{musical}]_x \cdot \theta_m + \mathbf{?}([\texttt{orfeo}]_x \wedge F^{-1}[\texttt{musical}]) \cdot \theta_o \\ \theta_m &= \mathbf{?}\ \texttt{gay}_i \cdot K_i\ \texttt{gay}_x \\ \theta_o &= \mathbf{?}\ \texttt{gay}_i \cdot (K_i\ \texttt{gay}_x \wedge K_i\ K_x\ \texttt{gay}_i) \\ &\text{with } x \text{ and } i \text{ arbitrary agents} \in \mathbf{I}\end{aligned}$$

The reason why the operator for 'sometime in the past' is used, as opposed to the 'in previous step' is because this allows for a bit more variability in the situation. In a real conversation, it is believable that Anne might have a response to the statement too, relating to something with music. If that is the case, Jane's response will not be in the next state after the first statement. The acknowledgement not being in the first state following does not mean that it is not an acknowledgement to the first statement. The current way of describing this formally allows for this option.

In the model, it would be that after installation of the protocol, and announcements of `musical` and `orfeo`, both Kate and Jane will know that the other is gay.

$$\mathcal{M}, s \vDash [!\eta][\texttt{musical}]_k\ [\texttt{orfeo}]_j\ (K_j\ \texttt{gay}_k \wedge K_k\ \texttt{gay}_j)$$

Additionally, Kate will also know that Jane knows that Kate is gay, due to the `orfeo` statement. The full model will then look like this:

$$\mathcal{M}, s \vDash [!\eta][\texttt{musical}]_k\ [\texttt{orfeo}]_j\ (K_j\ \texttt{gay}_k \wedge K_k\ \texttt{gay}_j \wedge K_k K_j \texttt{gay}_k)$$

In addition to the knowledge operators added in the previous example, it became necessary to add temporal operators into the formalisation. The only operator used is the $F^{-1}\varphi$, that says that φ has been true at some point in the past.

5.3 Diplomatic Situation Formalisation

As this is by far the more complicated example, the formalisation will be most complex as well. It will still have the same set-up of agents, propositions, actions, valuations, postconditions, and a protocol. However, some of the reasoning within the example will not be described completely formally, as it is quite complex.

There will be two agents, Alice and Bob. The board will be a three by three board, with chips in three different colours. The smaller board is to save on computations later. Even though it is smaller than in De Weerd et al. (2017), it should still be able to display the proper level of complexity for this example. Each agent will have three chips, each of which are one of the three available colours. One agent will have a goal at two steps distant, the other agent's goal will be at three steps distant. Because both agents have 3 chips, this will result in one chip being superfluous. It can still be bartered over, and the agent ending with an extra chip will also gain points for this.

Aside from this specific information, we need a way to describe the rules of the game, the negotiation of the game, and the board and chips. This section will work through those three elements in order, before combining them into one.

5.3.1 Rules of the Game

This game (as any game) has a start and an end. It also has a board of size $n \times n$, i agents, z chips, and c colours. The agents need to reach their goals, which can be the same, or different. Additionally, the distance to the goal can differ per agent. Agents are to aim for gaining as many points as possible. These points are determined as explained in Section 3.3.

The negotiation process goes as follows: one of the agents starts by making an offer (consisting of a chip-distribution). The other agent can accept this offer, reject the offer and make a counteroffer, or reject the offer and end negotiations. Accepting the offer also ends negotiations.

In terms of what they are aiming for, agents will have different sub-goals: reaching the goal, having as many chips left as possible, and ending negotiation as quickly as possible. Because of the point distribution, these will be prioritised.

Formally, the actions in this model are the negotiation offers and accepting or refusing of these. The protocol, stated informally, will be: as long as the game has not terminated, make an offer that is advantageous to the agent. This does not completely take the turn-taking aspect into account, or the fact that a game needs to start. Furthermore, somehow the choice the agent makes for an offer needs to be a bit flexible, as agents will not reach a sensible conclusion if they keep making the same offers again and again.

A revised version of the protocol, and closer to an actual formal one, is: if the game has not started, start it; then do offer actions until the game is terminated.

The first few definable elements of this problem are the agents and the actions.

$$
\begin{aligned}
\mathbf{I} &= \{\text{Alice}, \text{Bob}\} \\
\mathbf{P} &= \{\texttt{start}, \texttt{end}, \texttt{turn}\} \\
\mathbf{\Sigma} &= \{\texttt{openingOffer}^{\texttt{x}}_{\texttt{i}}, \texttt{accept}_{\texttt{i}}, \texttt{rejectOffer}^{\texttt{x}}_{\texttt{i}}, \texttt{rejectEnd}_{\texttt{i}}\} \\
V &: \{\texttt{start} = 0, \texttt{end} = 0, \texttt{turn} = 1\}
\end{aligned}
$$

The actions `openingOffer` and `rejectOffer` will take as argument x the offer made. These offers will be a chip distribution. For example, if the colours used are red, green, and blue,

shortened to *r, g*, and *b*, and there are two chips of each colour, an offer could look like: $\texttt{openingOffer}_{A}lice^{ggr.bbr}$. The colours will be ordered alphabetically to remove duplicate examples and for ease of reasoning. This would be an example of Alice performing the opening offer to start the game, and proposing that he gets two green chips and one red one, leaving two blue and one red chips for Bob. This will be necessary at a later stage when the postconditions are discussed. All arguments have an i to denote which agent performed that action. The atomic propositions *start* and *end* are boolean to denote whether the game is started and ended, and the proposition *turn* defines which agent's turn to offer it is.

5.3.2 Playing Strategies

The goal of each agent is to have the highest possible score at the end of negotiations. Worth noting is that highest possible is defined as competitively against self, and not against the opponent. Even while trying to gain a higher score, it will often be necessary to make an offer that will be of benefit to the other agent so they are more likely to accept it.

The score could be done as some element of postconditions attached to the actions. Another option is to take this part of the program away from the formal structure, and look towards the implementation element, as previously done in De Weerd et al. (2017). The downside to taking that approach is that it makes this formalisation less complete than the others.

Another aspect of strategy is which path they choose. To properly reason about that, some temporal logics are needed. This works both for ensuring that the agents take turns according to the rules of the game, and for making estimations of what the other agent is likely to agree to. More complex strategies can be thought out when taking into account the history of the negotiations.

This aspect of this example will not come into play until extra reasoning is added to the agents.

5.3.3 The Playing Board

The board will be three by three tiles in size, and each agent will hold two chips. The board size is smaller than described in the example description. This choice was made because in the implementation stage, calculating all possible options for the distribution of the chips will take up more computation power. In the smaller board, there are fewer paths possible, so it will be easier to reason about the set of paths. To retain some difference between paths, the starting point will be in the bottom central space. The possible goal spaces are those reachable in two or three steps. This can be seen in Figure 5.2.

The space denoted by 'S' is the starting point, 'X' denotes goals reachable in 2 steps, and 'Y' denotes goals reachable in 3 steps. The agents will have 3 chips each, and there will be a chip distribution such that both agents will be able to reach he goal if negotiations are successful. The tiles will have one of two or three colours. For two colours, there are fewer options on the paths, so it should be easier to reach goals. However, it would be more difficult to predict what goals the other agent is aiming for. Alternatively, with three colours, reaching the goals becomes more difficult (depending on the choices made for chips in the game), but because paths will be more unique, it will be easier to estimate which goals the other agent could be aiming for. In Figure 5.3 two example boards are shown, with two and three colours. Both colour distributions were determined randomly. The goals for Alice and Bob are determined by the red 1 and 2 shown in the example. Alice's goal is the '1', and Bob's is the 2.

The final choice for colours is the three colour board. The set of chips available in the game needs to be enough to let both agents reach their goal, with one chip remaining.

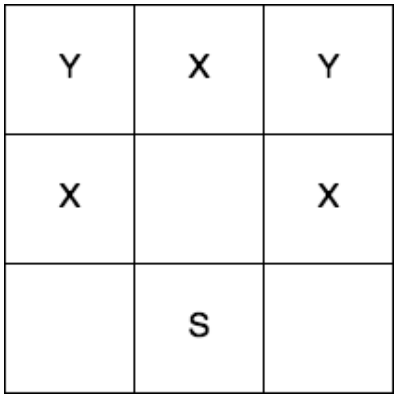

Figure 5.2: *The smaller coloured trails example with start and goal states indication.*

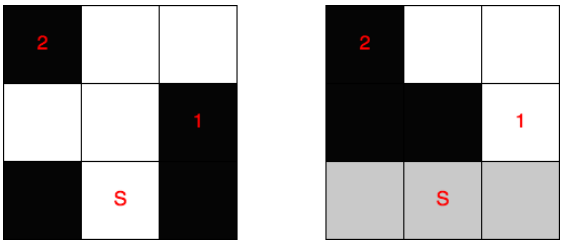

Figure 5.3: *A coloured in coloured trails example.*

5.3.4 Paths to Goal

For Alice there are *two* possible routes to the goal, each of length two. For Bob there are *three* possible routes to the goal. The different routes, and which colour chips that would cost, are shown in Table 5.1. In the colours reference column, the colours are mentioned in alphabetical order, for ease in comparing options at a later stage.

agent	paths	colours	reference	ref colours
Alice	up, right	black, white	1:ur	bw
Alice	right, up	grey, white	1:ru	gw
Bob	left, up, up	grey, black, black	2:luu	bbg
Bob	up, left, up	black, black, black	2:ulu	bbb
Bob	up, up, left	black, white, black	2:uul	bbw

Table 5.1: *Possible routes for the agents given the playing board, and the colours those routes would take.*

All separate options are found by taking all possible combinations for Alice with all possible combinations for Bob. An example of a route combination is when Alice takes the route: *up, right*, and Bob takes the route *left, up, up*. This is notated as 1ru-2luu. In these notations, Alice is referred to as agent 1, as her name is first when ordered alphabetically. Bob is agent 2. All combinations of chips necessary for cases in which both agents reach their goal are shown in Table 5.2.

This table also shows that even though there are six different route combinations, there are

combination	colours	shorthand
1ur-2luu	black, white, grey, black, black	bbbgw
1ur-2ulu	black, white, black, black, black	bbbbw
1ur-2uul	black, white, black, white black	bbbww
1ru-2luu	grey, white, grey, black, black	bbggw
1ru-2ulu	grey, white, black, black, black	bbbgw
1ru-2uul	grey, white, black, white, black	bbgww

Table 5.2: *The colours of the chips needed for all possible routes when both agents reach the goal.*

only five unique chip-colour combinations for these routes. Because the aim is to create a situation in which both agents have three chips, one extra chip should be added to the combination chosen. In adding a chip, this will likely cause other combinations to be possible as well, leading to a situation in which there are multiple resolutions of the game possible in which both agents reach their goal.

The combination of colours chosen is bbgww+g. This is the combination needed for 1ru-2uul, plus an extra grey chip, which also makes 1ru-2luu possible. In this case the superfluous chip would be white. With two different options for resolution, it should be interesting for agents to bargain, without making it too easy. The starting distribution of the chips is randomly determined as follows. Alice receives a white chip, a grey one, and a black one, and Bob receives a grey one, a white one, and a black one. Alice will be able to reach her goal with this, but Bob will need to negotiate for a black chip. A different option is to forcibly make sure no agent can reach its goal with the starting chips, in which case the distribution would be: 1: black, black, grey, and 2: grey, white, white.

5.3.5 Formal Definitions

With the rules of the game, playing strategies, playing board, and paths to the goal all defined, it is possible to formalise this example.

Because the example used here is of a different format than the one used for formalisation in De Weerd et al. (2017), the way postconditions are defined is also slightly different from how it is done there. What they do is that their postconditions are definitions of how the value of certain atomic propositions are changed by executing an action. In the previous two examples the postconditions were mostly changes to specific agents' knowledge about the environment and the other agents.

In this example there are the atomic propositions that determine whether the game has started, whether the game has ended, and which agent is next to perform an action. Additionally, there will be updates of knowledge, as in the previous examples. A summary can be found in Table 5.3.

$\texttt{openingOffer}_i^x$	start=1
$\texttt{accept}_i$	end=1
$\texttt{rejectEnd}_i$	end=1
action_i	if i=1, turn=2. if i=2, turn=1
$\texttt{openingOffer}_i^x$	update knowledge with current offer x
$\texttt{rejectOffer}_i^x$	update knowledge with current offer x

Table 5.3: *The postconditions of Example 3. In this table 'action' is a placeholder for all of the actions defined above.*

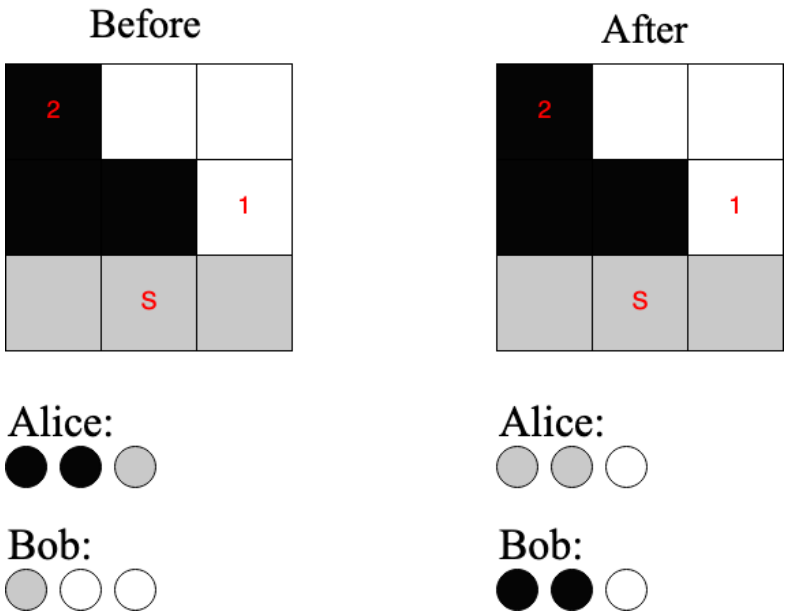

Figure 5.4: *The game board with chips at the begin and end of the game modelled.*

As stated above, the protocol will be 'if the game has not started, start it; then do offer actions until the game is terminated.' The only element not yet defined is which set of chips to offer. This choice will be determined by a comparable algorithm to what was used in De Weerd et al. (2017), and will therefore not be discussed in this section. The knowledge update will be none initially. As the example or implementation gets more complex, the knowledge involved in reasoning will also become more complicated.

$$\eta = ?\neg\texttt{start} \cdot \texttt{openingOffer}_1^x + (?\neg \mathit{fin} \cdot \text{action}_{\texttt{turn}})^*, \text{ where:}$$
$$x \text{ is chosen by the offer algorithm}$$
$$\mathit{action} = \texttt{rejectOffer}^x \vee \texttt{rejectEnd} \vee \texttt{accept}$$

To show a model example, one single possible run through the game should be chosen. This is difficult to do realistically, as the offers will eventually be chosen by the algorithm, and it is difficult to predict on what basis an agent would accept or reject an offer. Furthermore, the score the agents reach in playing this is currently not formally expressed. Take an example where the opening offer is that the chips distribution stay the same. After all, agent 1 can currently reach their goal, and will even have one chip remaining. Agent 2 will reject this offer, and perform a counteroffer that enables agent 2 to reach their goal. This counteroffer is 1:ggw, 2:bbw. Coincidentally, this is a distribution that agent 1 can also reach their goal with, so they accept.

$$\mathcal{M}, s \vDash [!\eta]\langle \texttt{openingOffer}_1^{1:bgw,2:bgw} \cdot \texttt{rejectOffer}_2^{1:ggw,2:bbw} \cdot \texttt{accept}_1\rangle$$
$$(\neg\langle \mathit{end}\langle\top \wedge \langle \texttt{openingOffer}_1^{1:bgw,2:bgw} \cdot \texttt{rejectOffer}_2^{1:ggw,2:bbw} \cdot \mathit{accept}_1\rangle\langle \mathit{end}\rangle\top)$$

If the situation is started with a chip distribution from which neither agent can reach the goal, the reasoning is slightly more interesting. The starting configuration is then 1:bbg, 2:gww.

Recall that Alice needs bw or gw to achieve their goal, where Bob needs bbb, bbg, or bbw. Say Alice opens with the offer of 1:bgw, 2:bgw. With this, Alice would be able to reach the goal. As Bob still cannot reach his goal, he will refuse, and do counteroffer 1:ggw, 2:bbw. As Alice can also reach her goal with this distribution, she will accept and the game ends.

The game board and chip distribution at the begin and end of the negotiation can be seen in Figure 5.4.

In the current formalisation of the model, there are no changes to the formal model.

$$\mathcal{M}, s \vDash [!\eta]\langle \texttt{openingOffer}_1^{1:bgw,2:bgw} \cdot \texttt{rejectOffer}_2^{1:ggw,2:bbw} \cdot \texttt{accept}_1\rangle$$
$$(\neg\langle end\langle \top \wedge \langle \texttt{openingOffer}_1^{1:bgw,2:bgw} \cdot \texttt{rejectOffer}_2^{1:ggw,2:bbw} \cdot \texttt{accept}_1\rangle\langle end\rangle \top)$$

There were some additions made to the existing formalisation to be able to incorporate all necessary elements in this description. In addition to the knowledge operators used, there are some aspects of this formalisation that do not translate easily into either formal logic or the existing protocol-oriented formalisation. This is mostly the use of external functions to determine which offers are made and accepted, and the scoring system. Additionally, there is some variability in the postconditions. Where the postconditions in De Weerd et al. (2017) are all atomic propositions, and in the previous examples they were all knowledge updates, in this example it is a combination of both.

Chapter 6

Simulation

This chapter concerns the design and implementation of the simulation created. Modelling the example situations in the simulation will be discussed in the next chapter, Chapter 7.

First the general design of the simulation will be discussed, followed by a description of how this design was realised in an implementation.

6.1 Design

The simulation is comprised of several elements that act in concert with one another. The first of these is the general system that will run and set up the simulation. This includes the framework for storing all information about the agents and protocols.

The second element is the parser. The different examples in the simulation will be modelled by reading them in as input. The parser is called by the system and ensures that the input is transformed into the structure usable by the rest of the system.

The third part is the logical framework of the simulation. It contains the logical structures as well as a means to store logical information. As it is necessary to be able to reason about the knowledge an agent has, or about which action brings an agent closer to the goal, a logical framework is needed to perform these actions.

6.2 General System

The general system is the core of the simulation. The most important part is the central system, which runs the simulation. It is in charge of interpreting the input, setting up the system for reasoning, and running the simulation. Interpreting the input will be discussed in Section 6.3.

Setting up the system means that after the input has been interpreted, all information is stored in a way that makes it easily accessible while running the system. There are agents, protocols, and actions. The agents are instantiated with all information they need to make decisions, as well as the information they start the simulation with. Part of this information they start with is beliefs on the protocols. This is further described in Sections 6.6 and 6.5.

To run the system, the central system first checks whether the preconditions of any of the protocols are met. If so, these are applied. The agents will then get asked to submit an action. The agents choose the action by determining the progress that action will help them in advancing on their goal.

These actions are acted out by the central system, which does this by updating the situation in the way instructed by the postconditions of the actions. To avoid possible conflict, the agents are asked for actions in a turn-based way. This way each agent chooses its action based on the most current state. If a turn-order is specified, that is followed, otherwise the agents are asked in order depending on the first letter of their name. After each action, the central system again checks whether any protocols are applicable.

The amount of rounds is specified per example. A round is defined a series of turns in which each agent has exactly one turn.

6.3 Parser

To transform each separate example into a structure in this simulation, we decided to formulate them as inputs. This led to creating a parser that reads an input containing all necessary information to describe the simulation. The input is then transformed into a shape the central system can use. The parser used can be seen in the appendix on page 63.

6.4 Logical Framework

Because there is logical reasoning involved in reasoning in these simulations and in reasoning about information, a logical framework was necessary. The one used here was created specifically for this thesis. The framework itself is simple, containing just the structure of a Formula, a Literal, and a Negation; and Conjunction, Disjunction, Knowledge, Implication, and Bi-implication operators. It further contains a $\top$ and a $\bot$, and can also simplify formulas.

Additionally, a Kripke model is used to be able to properly evaluate formulas. It is the knowledge operator for which this is essential. To function properly, the Kripke model uses a state map of the possible or available states, a representation of the relations in the model, and the agents in the model. The updating of knowledge will happen in two ways. First, if an agent gains knowledge, the relations connecting from states in which that knowledge is true to states in which it is not are removed. Secondly, if something is common knowledge among all agents, the states in which that information is false are removed, as well as all relations connecting to that state. The way in which these relations and states are removed is inspired by public and private announcements, as described in Van Ditmarsch et al. (2007, chapters 4 & 5)

The Kripke model is also used for agents to evaluate the consequences of their actions, used when choosing an action. Finally, it is used to determine whether agents have achieved their goals.

6.5 Agents

This simulation would not be an agent-based simulation if the agents did not have some agency. As described in Section 6.2, the agents choose which action to perform. They also have a goal they are working towards, which influences the actions they choose. The preference for agents is, of course, to make sure that the actions they perform lead them to reaching their goal. The agents test the actions in a copy of the Kripke model, to see the results of the action. If the consequences of the action lead to the agent reaching their goal, then that action gets high preference. If the action causes a new action to be possible - because its preconditions are now met - it gets a low preference. If neither happens, the action will get a zero score assigned. The

agent will then choose an action at random among the highest preference score that their actions have.

6.6 Reasoning

The reasoning in the simulation happens within the Kripke model in a few different ways. At the beginning of the simulation, agents can have knowledge. Typically, this will be critical information about themselves, or beliefs about the working of protocols. These initial beliefs are set in the Kripke model as belief updates. A belief update works as follows: for the agent that holds that belief, any relations to states wherein the belief is false are removed. This idea is based round the idea of private announcements as mentioned in Section 6.4, with a slight difference. Whereas private announcements involve removing the relations between states in which the valuation for the announcement was different, these belief updates only remove the relations in a single direction. Because these updates are belief updates, the knowledge mentioned in the simulation and in the Kripke model is not actual true knowledge, but rather, it relates to beliefs, some of which may be true. Beliefs for agents can be either atomic propositions (e.g. "The meeting is on the first floor") or beliefs about the protocols involved in a situation.

The changes in the model after this initial belief update for agents stem from actions and protocols. The protocols are chosen by the central system, based on what is possible to run, and the actions are chosen by the agents. Both the actions and the protocols affect the Kripke model through use of public announcements. These remove the states wherein the message of the announcement is false. Because of the belief updates done by the agents earlier, and their belief connections, removing these states will ensure that agents are aware of both the statement in the announcement, as well as their beliefs about the information transferred.

6.7 Implementation

This simulation was implemented in python, version 3.5.3. Additionally, the Lark library[1] was used to construct the parser. The full grammar used for the input can be found in Appendix A.

The specific inputs for the examples will be discussed in Chapter 7 and are printed in full in Appendix B.

The code for this thesis, together with a README, can be found at `https://github.com/lauravdbraak/Hidden_protocols_implementation`.

[1] Found at `https://github.com/lark-parser/lark`, page last accessed on 11/07/2019

Chapter 7

Examples in Simulation

The formalisations of the examples created in Chapter 5 have been input in the simulation described in Chapter 6. Here, we will show the discrepancy with our theorising and show the working of each example in the simulation.

7.1 Language Example

We will discuss the input for the language example (as introduced in Section 3.1 and formalised in Section 5.1) in sections, followed by showing how the reasoning works in this example, and the results for this example.

7.1.1 Input

The first two sections of the input show the literals that exist in this example, and what the true value for each is, if defined.

```
Vars:
  _first
  _ground
  _abeamerican
  _brittamerican
```

Here, literals start with an underscore to remove ambiguity from the input, so it can be read by a parser. These literals match the ones defined in Section 5.1, with slight adaptation to allow for better use in implementation. The literal `_first` refers to the statement 'The meeting is on the first floor', and `_ground` means: 'the meeting is on the floor on ground level'. The proposition `next` from the formalisation is now found as the negation of `_ground`, as they are semantically each other's negation.

The propositions related to the nationality of the agents had to be adapted a little more, as the logic used in the Kripke model is propositional, and therefore predicates cannot be used. So, intuitively, `_abeamerican` means that Abe is American, and `_brittamerican` means that Britt is American.

```
Truth:
  _first
  ~ _ground
  _abeamerican
  ~ _brittamerican
```

Regarding truth in this model, this concerns the true state at the end of the run: `_first` is going to be announced, so it is true. We know that Abe is American, and that Britt is not. For the run of the model it matters little whether or not we define `_ground` as true, but we chose to do that here for simplicity's sake.

```
Actions:

Protocols:
  announcefirst
    if _true
    then _first
```

The agents in this situation are not able to perform any actions. The situation is designed to be simple, so this is not included in the example. There is an outside announcement in this situation, which is framed as a protocol shown above. Protocols have a name, and an 'if' and a 'then' field. The precondition here allows the protocol to be announced at any time; there are no constraints to when this can be announced. The public announcement of `_first` is what will remove states from the model in such a way that they can reason about on what floor they deem the meeting to be.

```
Agents:
  Abe
    Info:
      _abeamerican
      (_first <-> _ground)
    Acts:
    Goal:
      ((Abe knows _ground) | (Abe knows ~ _ground))
  Britt
    Info:
      ~ _brittamerican
      (_first <-> ~ _ground)
    Acts:
    Goal:
      ((Britt knows _ground) | (Britt knows ~ _ground))
```

Lastly, there is the input that defines the agents. In general, agents have a name, information they believe in, actions they can perform, and a goal. The protocols as described in the formalisation are most clearly expressed as agent beliefs. In this case, the agents have beliefs on the location of the first floor. For Abe, because he is American, he believes that the first floor is the ground floor of the building. So if he is told a meeting is on the first floor, he believes the meeting is on

the ground level of a building. Similarly, for Britt if she is told a meeting is on the first floor, she believes the meeting is on the next level up from the ground level in the building. Their goals are the same: they want to know on which floor the meeting is, expressed as above.

Finally, the example is started with a configuration file:

```
lang_config = {'title' : "language", 'agent_names': ["Abe",
    "Britt"], 'turns' : [], 'rounds' : 1}
```

The title ensures that the full input file can be found and read. The names of the agents are provided separately for ease of setting up. Turns (not used here) can be used to determine a playing order between the agents. This is used only if the agents have actions to perform. The default is alphabetical, but that is overruled if one or more of the agents are included in this configuration file. If not all of the agents are mentioned in the turn field, the agents not mentioned will have their turn after the ones named, and will be ordered alphabetically. Finally, rounds refers to the number of rounds that are played. Each round consists of the system executing protocols, and each agent being asked for an action once. As there is only one protocol present, and no actions, it is not of interest to play for longer, so this is set to 1.

7.1.2 Reasoning and Results

In this example, the reasoning is fairly simple. At the set up, both of the agents learn their nationality and their beliefs about the location of the first floor. This will eliminate a number of relations in the Kripke model for both agents. Then, when the announcement `_first` has been done, the worlds in which `_first` is not true will be removed. As a consequence of this, the only relations remaining will always point to worlds still considered possible for each agent, and worlds with known false information will not even be taken into account in this evaluation. The results for this example are then as follows:

```
Performed actions: []
True state: 10, with values: {'_ground': False, '_first': True,
    '_abeamerican': True, '_brittamerican': False}

Agent Abe, has goal  (#_Abe _ground |  #_Abe ~ _ground), which is
    achieved
Agent Abe believes that the statement: 'the meeting is on the
    ground floor' is True.

Agent Britt, has goal  (#_Britt _ground |  #_Britt ~ _ground),
    which is achieved
Agent Britt believes that the statement: 'the meeting is on the
    ground floor' is False.
```

We give a brief explanation to go with this output. There have been no actions performed in this run. This is sensible, as there were none there to begin with. The true state from which the final evaluation was done is printed. This is not relevant here, where there is only one true world, but it will be more important in the social example, discussed in the next section. We also see that both agents achieve their goals. Additionally, we can see that Abe believes the meeting to be on the ground floor, where Britt believes this to not be the case. As we defined

the negation of `_ground` to mean 'the meeting is on the next floor up from the ground floor', we can see that Britt believes the meeting to be on the next floor up from the ground floor. This is both as described in the formalisation.

7.2 Social Example

As in the previous section, we will first discuss the input for this example, followed by a description of the reasoning and the results. This example was introduced in Section 3.2 and formalised in Section 5.2.

7.2.1 Input

The set up of the input is much the same as that of the previous example, though the content differs.

```
Vars:
  _gaykate
  _gayjane
  _gayanne
  _musicaljane
  _musicalkate
  _musicalanne
  _orfeojane
  _orfeokate
  _orfeoanne
```

Because the decision was made to use propositional logic, the set of literals is more extensive than seemingly necessary, and also more extensive than we had described in the formalisation. The literals `_gaykate`, `_gayjane`, and `_gayanne` intuitively denote whether the agent named is gay. Because actions and protocols result in formulas being announced, they need to have their own literal to announce. The literal `_musicaljane` means that Jane says 'I am musical, I like the sound of Kathleen Ferrier's voice', and happens in much the same way for Kate and Anne. The literal `_orfeojane` means that Jane says 'I agree, I especially liked her portrayal of Orfeo in Orfeo and Euridice'. For ease in reading, these statements will be described as 'Jane says `musical`' and 'Jane says `orfeo`' in the rest of this section.

```
Truth:
  _gaykate
  _gayjane
  ~ _gayanne
  ~ _musicalanne
  ~ _orfeoanne
```

The truth is a bit more complicated in this example. Because, as we will see later, which actions the agents perform is not deterministic, so it is not possible to determine a true world from the start with the guarantee that this world will still be the true world at the end. Instead, the code can deal with the possibility of multiple possible true worlds. Truth will be evaluated from that world as if it is the (only) true world, but if the world is removed (during a public announcement), a new true world is chosen. All these true worlds will have the above valuations in common.

As determined when the example was introduced, Kate and Jane are gay, Anne is not. Because Anne is not gay, she does not have the protocols relating to being able to make announce-

ments that she is musical, or that she liked the Orfeo performance particularly. These literals still exist as possibilities, because the other agents do not know that these are not possible for Anne, so they can be considered possible. However, these are known to be not true, which is why they are labelled as `False` in the list of truth about literals.

```
Actions:
  musicaljane
    pre _gayjane
    post _musicaljane

  musicalkate
    pre _gaykate
    post _musicalkate

  orfeojanek
    pre _musicalkate
    post (_musicaljane & _orfeojane)

  orfeojanea
    pre _musicalanne
    post (_musicaljane & _orfeojane)

  orfeokatej
    pre _musicaljane
    post (_musicalkate & _orfeokate)

  orfeokatea
    pre _musicalanne
    post (_musicalkate & _orfeokate)

Protocols:
```

In this situation, there are no protocols in the sense that there are no outside statements releasing information to the agents. Contrary to the language example, there are actions that the agents can choose. These actions, similarly to the protocols, have a name and a precondition and a postcondition. Again, because it is not possible to use quantifiers, there are multiple options listed. The first two are similar: at any time, when asked for an action, Jane and Kate can say that they are musical. This is because each of the agents knows that she herself is gay, and all that is required to make such a statement is the knowledge that she is gay.

The orfeo actions are slightly more complex, because they are said only in response to a `musical` action. As such, they have that `musical` statement as precondition. The `orfeo` statement in the initial example is both an acknowledgement and a piece of additional information. As such, we have chosen to model the postconditions of the action as if they specifically say 'I am musical too, I especially liked her portrayal of Orfeo in Orfeo and Euridice'. As these actions are a response, it is possible to see in the name of the action to whom they are a response by the initial at the end of the name. For example, 'orfeokatej' means: Kate says `orfeo` as response to Jane.

```
Agents:
  Kate
    Info:
      _gaykate
      (_musicaljane -> _gayjane)
      (_musicalanne -> _gayanne)
      (_musicalkate -> _gaykate)
    Acts:
      musicalkate
      orfeokatea
      orfeokatej
    Goal:
      ((((Kate knows _gayjane) & (Kate knows _orfeojane)) | (Kate
         knows ~ _gayjane)) | (((Kate knows _gayanne) & (Kate
         knows _orfeoanne)) | (Kate knows ~ _gayanne)))
  Jane
    Info:
      _gayjane
      (_musicalkate -> _gaykate)
      (_musicalanne -> _gayanne)
      (_musicaljane -> _gayjane)
    Acts:
      musicaljane
      orfeojanea
      orfeojanek
    Goal:
      ((((Jane knows _gaykate) & (Jane knows _orfeokate)) | (Jane
         knows ~ _gaykate)) | (((Jane knows _gayanne) & (Jane
         knows _orfeoanne)) | (Jane knows ~ _gayanne)))
  Anne
    Info:
      ~ _gayanne
    Acts:
    Goal:
      _true
```

Of the three agents, Anne is most simple, and Kate and Jane are identical (if we switch out the names). We will discuss Anne first.

As explained when the example was introduced, Anne is unaware of the information contained in the conversation around her. As such, she also does not have any actions she can perform. This does not necessarily mean that Anne is not allowed to say anything in the conversation, more that as she cannot say anything that will communicate information relating to the goals of the other agents, she does not have actions within this model. Regardless of whether she knows what the subtext of the conversation around her is, she knows that she is not gay, so that is still contained in the model. She also has no goal relating to the topic being modelled, hence her goal being set to $\top$.

Kate and Jane have some additional initial beliefs. Aside from the knowledge of their own gayness, they also have the belief that if an agent says she is musical, then that agent is gay. They both have three possible actions, mentioned only by their names here to decrease the length

of the input and increase the readability for the user. They can say they are musical, and give an `orfeo` statement in response to a `musical` statement from one of the other actions.

Their goal can be broken down into sections. For each of the other agents, they wish to know if they are gay or not. If they are gay, they also wish to receive an acknowledgement to show that the knowledge about being gay has been clearly shared between the two. This acknowledgement is in the form of an `orfeo` statement.

Finally, the configuration input for the social example is as follows:

```
soc_config = {'title' : "social", 'agent_names': ["Kate", "Jane",
   "Anne"], 'turns' : ["Kate", "Jane", "Anne"], 'rounds' : 2}
```

The title and names are self-explanatory. In the original explanation of the example, Kate is shown to start with a message, after which Jane responds. This is input into the system, so that will be the order in which they get asked for actions. Each agent has one turn per round. As Kate's and Jane's goals also refer to having received an `orfeo` statement from the other, there are two rounds in this simulation. In this way both agents have an opportunity to say `orfeo`, as well as `musical`.

7.2.2 Reasoning and Results

At the beginning of the simulation, the agents are each informed of their beliefs. Kate and Jane now know that they are gay, and Anne knows that she is not gay. Furthermore, Kate and Jane know that if an agent says she is musical, that means that she is gay.

Next, the agents are asked for an action. Kate is first, and she sees that `_musicalkate` is her only option, because no other agent has made a `musical` statement yet. So she returns that action, and a public announcement is made. Because Jane already knew that if an agent said she was musical, that meant she was gay, only relations to states with both `_musicalkate` and `_gaykate` or neither of those were left for Jane. When the worlds where `_musicalkate` is False are removed, Jane is left with only connections to worlds in which `_gaykate` is True, so she now knows that Kate is gay. Anne is oblivious to this, and has no change in her knowledge.

Jane is next. She now has two options, she can give her own `musical` statement, or respond to Kate with an `orfeo` statement. A `musical` statement will allow Jane to respond with `orfeo` next, as well as the `musical` statement that was already possible. The `orfeo` statement will do the same (as the acknowledgement is read as an 'I am musical too'), as well as give Kate the information on `orfeo` for Jane. For an ideal run, both agents need to say `orfeo`. Additionally, this implementation is under the assumption that not only are both Kate and Jane aware of this protocol, they are also driven enough towards their goal to only say relevant statements.

Because the goal is framed as: wanting to know if the other is gay, and wanting an acknowledgement that they know you are gay, the agents are not performing advanced reasoning towards the goal. They cannot reason about the value of saying `orfeo` versus `musical` because they make no assumptions regarding the goal of other agents, let alone if they want to be cooperative or competitive. They also do not reason about actions that the other agent may or may not be able to perform. This results in the function that gives the actions a score relating to how performing it gets the agent closer to the goal returning the same value for both actions.

This means that the agents will, after Kate has had the first turn and said `musical`, choose arbitrarily between `musical` and `orfeo`. In all cases, Kate and Jane will believe each other to be gay, and Anne to not be gay, and Anne will believe none of the agents to be gay. Whether Kate and Jane actually achieve their goal is dependent on whether both of them also say the `orfeo`

statement, and so can differ from run to run.

The results are then as follows:

```
Performed actions: [['musicalkate', 'Kate'], ['musicaljane',
   'Jane'], ['orfeokatej', 'Kate'], ['orfeojanek', 'Jane']]
True state: 438, with values: {'_orfeokate': True,
   '_musicaljane': True, '_gayanne': False, '_musicalanne':
   False, '_musicalkate': True, '_gaykate': True, '_orfeoanne':
   False, '_orfeojane': True, '_gayjane': True}

Agent Jane has goal ((( #_Jane _gaykate & #_Jane _orfeokate) |
   #_Jane ~ _gaykate) | (( #_Jane _gayanne & #_Jane _orfeoanne)
   | #_Jane ~ _gayanne)), which is achieved
Agent Jane believes that:
   _gayjane is True
   _gaykate is True
   _gayanne is False.

Agent Kate has goal ((( #_Kate _gayjane & #_Kate _orfeojane) |
   #_Kate ~ _gayjane) | (( #_Kate _gayanne & #_Kate _orfeoanne)
   | #_Kate ~ _gayanne)), which is achieved
Agent Kate believes that:
   _gayjane is True
   _gaykate is True
   _gayanne is False.

Agent Anne has goal (_true), which is achieved
Agent Anne believes that:
   _gayjane is False
   _gaykate is False
   _gayanne is False.
```

This is a run in which both agents make a `musical` statement and an `orfeo` statement. Because there are many true worlds, the one the final reasoning was performed from is printed, together with its valuations. It does not matter from which of the possibly true worlds evaluation is done, they will all result in the same results. The uncertainty that causes multiple worlds still to be considered true is because of the possible announcements that were not performed. In this case, there are none, but if one of the agents does not make a musical or an `orfeo` statement, the true value of those literals is still unknown at the end of the simulation.

We can see that, because both Kate and Jane have done an `orfeo` statement, both of them have reached their goal. Furthermore, you can see what each agent believes about the gayness of both themselves and the other agents. Kate and Jane now know that they are both gay, and that the other is aware of this. Anne believes all agents are not gay, and has not gained any information during the run.

7.3 Diplomatic Example

This example was introduced in Section 3.3 and formalised in Section 5.3. As this example is a lot more complex and contains a lot more different elements than the other examples do, it fell outside the scope of this thesis to implement this example in the simulation. It is interesting to discuss how this example could have been translated to the implementation of the simulation.

We will first discuss how the game mechanics would translate, followed by a description of the reasoning in the system.

7.3.1 Game Mechanics

For this example, recall that there is a game board, chips in different colours, a start and goal location for each agent, and rules on allowable paths to reach a goal. This example contains more information than just the knowledge contained in a situation, and we need to find a way they can be modelled.

The game board and chips are the same for both agents, and it would not be useful to store this information in the Kripke model, as it would grow to unmanageable size. So to model this, it would be better to create a separate representation for the game board, stored where the agents would be able to reach it, and can use it to evaluate possible paths. For the chips, as they are also visible to all agents, we would do the same.

The rules for accepted paths to a goal are slightly different in use. Instead of storing these in a similar way to the game board, it would be better to use these to create a separate function for evaluating possible paths, in much the same way as separate actions are now evaluated.

The starting location will be the same for all agents, so they will not have to reason about this information. The goal, however, is the only real piece of knowledge that can be reasoned about, so this is the information that will be placed in the Kripke model, the possible goals for the separate agents. With the game board as defined in Figure 5.2, there are five possible goal locations for each agent, which leaves 25 states to represent each possible goal combination (including the ones where the agents have the same goal). This would keep the Kripke model small and therefore manageable. Because this is not implemented, it is difficult to say whether this would be enough for all the reasoning needed, especially once we want to combine the reasoning about the other agent's strategy with the information gained from the actions they choose.

7.3.2 Reasoning

Part of the reasoning of this example, as well as the rest of the epistemic set-up, will be similar in design to the other examples discussed. However, as this example contains more complex mechanics related to action choice, some additional features will need to be added.

The goals the agents are aiming to reach in the game itself are not epistemic ones: they want to achieve the highest number of points possible. However, there is still reasoning involved in the situation, and reasoning about the other agent's goal state is a useful element to take into consideration for any agent who wants to score well. So contrary to the other examples, the goal will not be a knowledge goal that can be tested, or even a goal that can be considered 'complete'. The goal, for both agents, is to get the highest score possible. This goal will influence the actions chosen by the agents, and will not be evaluated separately.

The beliefs of the agent are of two types. The first is the knowledge they have of their own goal. The second, of which they will have no information at the beginning, is the belief about the goal of the other agent. The belief itself will become more defined as the agents play and

perform actions, but the agents will have beliefs about how to use the information contained within an action statement to reason about goals.

The actions will remain similar to how they were defined in the formalisation as shown in Section 5.3. However, this will prompt a change in the logic formalisation. Where in the previous examples it was still possible to work with propositional atoms, this would no longer be doable, so the logic atoms of the simulation would need to be changed to predicate atoms. Aside from rewriting the atoms, this should not cause change for the other examples.

The actions also contain extra information in a protocol fashion. Where an action can be read as just the action it is, making an offer for a specific chip distribution, it is possible to use the information from that distribution to attempt to determine the goal of the opposing agent.

Action choice is again different from the previous examples. On an abstract level, the working is the same: the action that enables a goal to be reached gets the highest probability of being chosen. On a lower level there is a difference. Where the other examples contained goals related to knowledge, and the actions would help towards that knowledge, the goal here is not an epistemic one. The action chosen needs to be the one that yields the highest expected score for the game. Better still would be to have a utility function that also takes into account the probability that the other agent will accept the offer.

A good solution on the mechanics of this is to look at the code provided by De Weerd et al. (2017). This shows which rules they used in their models, and the exact utility function, determined in part by what the opposing agent has accepted or refused in previous sessions. An alternative is to play according only to an agent's own information and goals, and ignore the fact that the other agent will also be playing towards a goal. Furthermore, it is possible to use the beliefs that are collected about the goal of the other agent. These can be used to make an estimate of whether the other agent would accept the current offer.

An important aspect of modelling these situations is to take into account any strategies the agents might have. The agents might want to play cooperatively, or extremely competitively without regard for the fact that the other agent needs to agree with the offer, or might even choose to value the goal of the opposing agent over their own. The latter is not a rational strategy, but would be an interesting one to consider regardless.

These strategies would influence the utility function connected to choosing the actions. To implement this, we would need to implement the different strategies for evaluating actions, and then let the strategy determine how the action is chosen.

If there are different strategies implemented, this influences the beliefs the agents have on how to reason about the knowledge gained from the action of the opponent. They can assume the opponent is using the same strategy, or they can reason with keeping an open mind on the chosen strategy. They could also attempt to figure out which strategy the other agent is using. All of this reasoning about strategies is reasoning about the beliefs on the protocols that the other agents hold, which connects back to the reasoning in the previous situations.

Chapter 8

Discussion

The discussion of this thesis is split into two parts. First, we will discuss the theoretical formalisation of the examples. After that, the implementation in the simulation will be discussed.

8.1 Theoretical Working of Examples

When creating the theoretical models of the example situations mentioned, the following additions to the existing protocol languages were discovered to be essential: epistemic and doxastic reasoning, and some aspects of temporal logic.

8.1.1 Epistemic and Doxastic Reasoning

The epistemic reasoning is necessary to reason about the beliefs of the other agents. The epistemic logic contained in the formalisation by Van Ditmarsch et al. (2014) is used for epistemic updates to a situation, but not for reasoning about the other agents. So it is not only the fact that there need to be knowledge operators, but also reasoning about possible beliefs and awareness of protocols of other agents, that makes this addition necessary.

Aside from reasoning about knowledge, there were belief updates in the implementation. These beliefs pertain to the beliefs or inside information of the other agents. If we wanted to incorporate these beliefs into the formalisation as well, there would need to be a clear definition of in which cases belief is appropriate, and when knowledge should be used.

To stay true to all implications of an example situation modelled, belief as well as knowledge should be added to the formalisation.

8.1.2 Epistemic Test Operator

Another element that is essential to add is the epistemic test operator. As mentioned in Chapter 5, the reasoning in all these examples concerns checking whether an agent knows information before actions or protocols can be executed. Van Ditmarsch et al. only define a test operator for Boolean propositions. This means that without the addition of an epistemic test it is not possible to determine if agents know information.

8.1.3 Temporal Logic

The temporal logic would need to be added to be able to deal with temporal conditions on actions and protocols. As can be seen in the second example, sometimes the working of a protocol is dependent on some previous condition happening in a previous state.

In that situation, introduced in Section 3.2, it is the case that one statement needs to be announced before another statement has any value. If the situation takes place in a real world, it is possible that other actions not relevant to the reasoning occur in between, which affects the exact temporal operator that needs to be used. It is preferable to be able to determine that the statement has to have occurred within the last five action steps.

If the statement has not been said, the protocol does not hold any value, as mentioned before. Additionally, if it was too long in the past, the context of the conversation will have changed, and it will not be relevant to respond to it in that fashion any more.

8.1.4 Reasons for the Additions

One could ask why these extensions did not already exist in the existing formalisation. For the two additions those reasons are slightly different. The epistemic logic was most likely not included because it was not immediately needed to be able to formalise the type of protocol they wanted to formalise. They created the Epistemic Protocol Logic to be able to reason with protocols. So, to show what happens when a protocol exists and is executed. This thesis takes that as a background and makes the step to reasoning *about* protocols. This requires an extra layer of reasoning, that could involve both epistemic and doxastic logic.

The reason why they did not add the temporal logic is probably because adding temporal logics to such new logics is highly unusual. There are several ways of representing change in logic systems, and as they were already working with protocols and actions that could both cause epistemic and factual change, there was probably no reason to add temporal logic onto that. It would probably have not been of use in the situations they aimed at modelling.

However, adding these additions to Epistemic Protocol Logic would allow us to represent real world examples in a way that represents the knowledge within them more accurately.

8.2 Practical Implementation of Examples

In the building of the practical implementation, some elements were changed slightly from the exact formalisation as described in Chapter 5. This is because a simulation has different mechanics than formal logic does. This section will discuss several of those changes and their effects.

8.2.1 Kripke Models

The largest difference between the formalisation and the simulation is that the latter uses a semantic representation of the knowledge. The use of a Kripke model for representing the knowledge has impact on a few different levels.

Kripke models are reliable for modelling the reasoning that exists within a situation. They contain all the states and all the relations, which are only removed when directed to do so after an announcement. Creating a different type of system that can deal with reasoning in this way would be a case of reinventing the wheel, and would not yield a better solution.

It does, however, have a downside. Because it contains all relations and all states, the models can grow large. The simulation created for this project has an explicit representation of the

Kripke model. This can result in the need for large amount of space to store the model, or a long time to test any knowledge in the model.

For the examples implemented this is not a problem. Even for the diplomatic example, if designed as discussed in Section 7.3, this should not be a problem. This is due to how the information within that example is represented. It is difficult to make an estimate of what sizes model would still be computable, but model size increases exponentially with every agent or atom added. Adding two agents with related atoms to the second example, and adding two or three actions would probably already be too much to compute with ease.

If we wanted to work with larger examples, it would be better to create a way of storing the Kripke model that is not explicit. A good way to do this could be by using SMCDEL, as created by Gattinger. It is a symbolic model checker for DEL that uses Binary Decision Diagrams. Binary Decision Diagrams are smaller than Kripke Models, and therefore easier to store, but they still maintain the full expressibility that a Kripke model has.

8.2.2 Belief updates

As explained in Chapter 6, belief updates to the model now only happen at the beginning of a run of an example. This is the more realistic scenario, if the intention is to model a real world example as it would happen in the real world. Agents are already aware of their beliefs and protocols, so when they are presented with new information, they directly know the result of applying the protocols, because they already knew them. This affects the working of the reasoning in the simulation.

It works well in the situation where there is only one update happening, and the reasoning is on a lower level. Here, lower level refers to statements that are presented as fact, and are not n-th order beliefs or knowledge about the information spread. We show this in the working of the socio-linguistic example in Section 7.1.

Unfortunately, as not all real world scenarios are this simple, this solution does not always provide the best results. This can be seen in the other implemented example. In the knowledge of the agents there is no reference to any beliefs concerning the `orfeo` statement. Additionally, where intuitively the goal would be to gain knowledge about what other agents know about an agent being gay ($K_{Kate}K_{Jane}$ `gaykate`), the goal refers to them having said the `orfeo` statement (K_{Kate} `orfeojane`).

The meaning of these two is very close to each other, but they are not the same. It is, for example, possible to consider the existence of a different acknowledgement statement that would convey the same meaning as the `orfeo` statement. If the goal refers specifically to the `orfeo` statement needing to be said, then the agent's goal in the simulation would not be reached. However, if it were a real world situation, it is more probable that the agent desires to receive any acknowledgement, not specifically the `orfeo` one.

A solution that would result in a more accurate final representation would be to connect the belief updates to the protocols and actions. Currently, when a protocol or action is executed, a public announcement is made. Because the agents already have beliefs, they reach their conclusions. An option is to add the private belief updates to be a postcondition for the protocol or action. This way, a public announcement with the information is sent, followed by a belief update to the agents that have protocols on that information. The Kripke model is updated accordingly, and agents still leave with all the necessary information.

The second issue with the belief updates at the beginning, is that if they refer to n-th order knowledge, too many relations might be removed erroneously, resulting in faulty beliefs. It is unclear whether this would also change with the changing of the moment that beliefs are updated.

Another addition to the belief system could be adding the agent that does the announcement to the representation of the announcement in the model. This already happens in the formalisation in 5.2, where we mention which of the agents says that they are musical. Adding this together with quantifiers in the logic representation can make the size of the input smaller, as it will not be necessary to define each option by hand. Rather, it would only be necessary to give the `musical` statement, and define which agents can say that statement.

8.2.3 Reasoning about beliefs

The final discussion point for the practical implementation is that there is currently no reasoning about beliefs of the other agents. It was introduced as necessary for realistic modelling in the theoretical section, but is not implemented in the current simulation.

The agents currently reason about what they believe of other agents, and about receiving the acknowledgement, but they have no real beliefs on the beliefs of other agents, or what the others know of protocols. It would be interesting to add those elements and see whether the predictions made during the formalisation stage are proven true.

Chapter 9

Conclusions

At the beginning of this thesis we posed the question: Is it possible to extend the existing formalisation so that we can represent real world examples? Can we show this in a simulation?

We can now answer these questions with 'yes, mostly'. We took real-world examples and expressed them in Epistemic Protocol Logic, as shown in Van Ditmarsch et al. (2014). The examples chosen were of different levels of complexity to show how well different types of information could be expressed. While doing this, two additions were found necessary and useful to be able to express the examples in full. This answers the first part of the research question.

Next, we built a simulation to express the examples in, in such a way that it would be a similar representation of the examples as the formalisation gave. As discussed in Section 8.2, it is possible to show the working of the examples in a simulation. However, it would provide a better result if the simulation could be expanded as discussed in that section. The current simulation is more on a level with the existing formalisation than with the theoretical additions, and it would be interesting to look at the full complexity of the examples in the simulation.

9.1 Future Work

There are several avenues of research opened up by this thesis. The first is to extend the simulation is such a way that the beliefs of agents are updated during a run. This would then let them occur simultaneously with the actions that fulfil the preconditions of that protocol. As the knowledge protocols for agents is expressed through these belief updates, this would cause all knowledge updates for agents to happen at the same moment, ensuring that the agents are always as well informed as possible.

A second avenue of research is to generalise the simulation to further examples. The coloured trails game discussed in this thesis would be an option, though it will require some tweaking to the implementation of the simulation to be able to run such a different type of example as explained in Section 7.3. Generalising the simulation to other examples will give more basis for determining that this is a correct and working simulation for Epistemic Protocol Logic, and that the extensions designed theoretically are also useful to other examples than the three mentioned here.

There is another possibility of further research that takes a step in a different direction. This would be to extend the Epistemic Protocol Logic with the proposed additions formally. Doing this gives the proposed additions some validity.

Finally, an option would be to build a different simulation. As discussed in Chapter 6, the approach chosen in this thesis is to create an explicit representation of a Kripke model. This

does have benefits, but it also has downsides, as discussed in Section 8.2.1. A different option would have been to take a model checker approach. This would involve generating all possible computation trees for a situation, and determining under which conditions certain facts would be true. This would eliminate the uncertainty we have now around what actions agents choose: the results from a model would be the same, even if the actions chosen by the agents differed. This is because even though the results of the simulation now differ, each combination and ordering of actions chosen has a deterministic result.

Appendix A

Grammar

Listing A.1: *Grammar in Lark*

```
start: _NL* literals _NL* truth _NL* actions _NL* protocols _NL*
   agents _NL*

literals: "Vars:" _NL [_INDENT (LITERAL _NL)+ _DEDENT]
truth: "Truth:" _NL [_INDENT (expr _NL)+ _DEDENT]
actions: "Actions:" _NL [_INDENT action* _DEDENT]
protocols: "Protocols:" _NL [_INDENT protocol* _DEDENT]
agents: "Agents:" _NL [_INDENT agent+ _DEDENT]

agent: AGENT _NL [_INDENT info _NL? acts _NL? goal _NL? _DEDENT]
protocol: NAME _NL [_INDENT "if" expr _NL "then" expr _NL?
   _DEDENT]  _NL*
action: NAME _NL [_INDENT "pre" expr _NL "post" expr _NL? _DEDENT]

info: "Info:" _NL [_INDENT (sentence _NL*)+ _DEDENT]
acts: "Acts:" _NL [_INDENT (NAME _NL*)* _DEDENT]
goal: "Goal:" _NL [_INDENT sentence _NL* _DEDENT]

?sentence: expr
  | protocol

?expr: single_expr
  | "(" conjunction ")"
  | "(" disjunction ")"
  | "(" implication ")"
  | "(" biimplication ")"

?single_expr: LITERAL     -> literal
  | negation
  | "(" knows ")"

conjunction: (conjunction | expr) "&" (conjunction | expr)
disjunction: (disjunction | expr) "|" (disjunction | expr)
implication: (implication | expr) "->" (implication | expr)
```

```
biimplication: (biimplication | expr) "<->" (biimplication | expr)

negation: "~" expr
knows: AGENT "knows" expr

AGENT.2: UCASE_LETTER LCASE_LETTER+
LITERAL: "_" LCASE_LETTER+
NAME: LCASE_LETTER+

_NL: /(\r?\n[\t ]*)+/

%declare _INDENT _DEDENT
%import common.LCASE_LETTER
%import common.UCASE_LETTER
%import common.WS_INLINE
%ignore WS_INLINE
```

Appendix B

Input Files

Listing B.1: *Language example input*

```
Vars:
  _first
  _ground
  _abeamerican
  _brittamerican

Truth:
  _first
  ~ _ground
  _abeamerican
  ~ _brittamerican

Actions:

Protocols:
  announcefirst
    if _true
    then _first

Agents:
  Abe
    Info:
      _abeamerican
      (_first <-> _ground)
    Acts:
    Goal:
      ((Abe knows _ground) | (Abe knows ~ _ground))
  Britt
    Info:
      ~ _brittamerican
      (_first <-> ~ _ground)
    Acts:
    Goal:
      ((Britt knows _ground) | (Britt knows ~ _ground))
```

Listing B.2: *Social example input*

```
Vars:
  _gaykate
  _gayjane
  _gayanne
  _musicaljane
  _musicalkate
  _musicalanne
  _orfeojane
  _orfeokate
  _orfeoanne

Truth:
  _gaykate
  _gayjane
  ~ _gayanne
  ~ _musicalanne
  ~ _orfeoanne

Actions:
  musicaljane
    pre _gayjane
    post _musicaljane

  musicalkate
    pre _gaykate
    post _musicalkate

  orfeojanek
    pre _musicalkate
    post (_musicaljane & _orfeojane)

  orfeojanea
    pre _musicalanne
    post (_musicaljane & _orfeojane)

  orfeokatej
    pre _musicaljane
    post (_musicalkate & _orfeokate)

  orfeokatea
    pre _musicalanne
    post (_musicalkate & _orfeokate)

Protocols:


Agents:
  Kate
```

```
    Info:
      _gaykate
      (_musicaljane -> _gayjane)
      (_musicalanne -> _gayanne)
      (_musicalkate -> _gaykate)
    Acts:
      musicalkate
      orfeokatea
      orfeokatej
    Goal:
      ((((Kate knows _gayjane) & (Kate knows _orfeojane)) | (Kate
         knows ~ _gayjane)) | (((Kate knows _gayanne) & (Kate
         knows _orfeoanne)) | (Kate knows ~ _gayanne)))
  Jane
    Info:
      _gayjane
      (_musicalkate -> _gaykate)
      (_musicalanne -> _gayanne)
      (_musicaljane -> _gayjane)
    Acts:
      musicaljane
      orfeojanea
      orfeojanek
    Goal:
      ((((Jane knows _gaykate) & (Jane knows _orfeokate)) | (Jane
         knows ~ _gaykate)) | (((Jane knows _gayanne) & (Jane
         knows _orfeoanne)) | (Jane knows ~ _gayanne)))
  Anne
    Info:
      ~ _gayanne
    Acts:
    Goal:
      _true
```

Bibliography

Ågotnes, T., van Ditmarsch, H., and Wang, Y. (2018). True lies. *Synthese*, 195(10):4581–4615.

Baltag, A. (2002). A logic for suspicious players: Epistemic actions and belief-updates in games. *Bulletin of Economic Research*, 54(1):1–45.

Baltag, A., Moss, L. L. S., and Solecki, S. (1998). The logic of public announcements, common knowledge, and private suspicions. *Proceedings of the 7th conference on Theoretical Aspects of Rationality and Knowledge, TARK'98*, (June 2015):43–56.

Carston, R. (1999). Using language. *Journal of Linguistics*, 35:167–222.

De Weerd, H., Verbrugge, R., and Verheij, B. (2013). How much does it help to know what she knows you know? An agent-based simulation study. *Artificial Intelligence*, 199-200:67–92.

De Weerd, H., Verbrugge, R., and Verheij, B. (2015). Higher-order theory of mind in the Tacit Communication Game. *Biologically Inspired Cognitive Architectures*, 11:10–21.

De Weerd, H., Verbrugge, R., and Verheij, B. (2017). Negotiating with other minds: The role of recursive theory of mind in negotiation with incomplete information. *Autonomous Agents and Multi-Agent Systems*, 31(2):250–287.

Dykstra, P., Elsenbroich, C., Jager, W., Renardel de Lavalette, G., and Verbrugge, R. (2013). Put your money where your mouth is: DIAL, a dialogical model for opinion dynamics. *Journal of Artificial Societies and Social Simulation*, 16(2013):1–31.

Dykstra, P., Jager, W., Elsenbroich, C., Verbrugge, R., and Renardel de Lavalette, G. (2015). An agent-based dialogical model with fuzzy attitudes. *Journal of Artificial Societies and Social Simulation*, 18:1–14.

Fagin, R., Halpern, J. Y., Moses, Y., and Vardi, M. Y. (2003). *Reasoning about Knowledge*. MIT Press, Cambridge, 2nd edition.

Gal, Y., Grosz, B., Kraus, S., Pfeffer, A., and Shieber, S. (2010). Agent decision-making in open mixed networks. *Artificial Intelligence*, 174(18):1460–1480.

Gattinger, M. (2018). Smcdel - an implementation of symbolic model checking for dynamic epistemic logic with binary decision diagrams. Version 1.0.0.

Gaudou, B., Herzig, A., Lorini, E., and Sibertin-Blanc, C. (2011). How to do social simulation in logic: Modelling the segregation game in a dynamic logic of assignments. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 59–73. Springer Berlin Heidelberg.

Grice, H. P. (1975). Logic and conversation. In Cole, P. and Morgan, J., editors, *Syntax and Semantics*, volume 3, pages 41–58.

Grice, P. (1989). Further notes on logic and conversation. In Cole, P. and Morgan, J., editors, *Studies in the Way of Words*, pages 41–57. Harvard University Press.

Grossi, D., van der Hoek, W., Moyzes, C., and Wooldridge, M. (2016). Program models and semi-public environments. *Journal of Logic and Computation*. exv086.

Grüne-Yanoff, T. and Weirich, P. (2010). The philosophy and epistemology of simulation: A review. *Simulation and Gaming*, 41(1):20–50.

Hofstede, G. and Hofstede, G. J. (2005). *Cultures and Organizations Software of the Mind*. McGraw-Hill, second edition.

Huth, M. and Ryan, M. (2004). *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge University Press.

Meyer, J.-J. and van der Hoek, W. (2004). *Epistemic Logic for AI and Computer Science*, volume 41. Cambridge University Press.

Parikh, R. and Ramanujam, R. (2003). A knowledge based semantics of messages. *Journal of Logic, Language and Information*, 12(4):453–467.

Singh, S. (1999). *The Code Book*. Doubleday, New York.

Teepe, W. (2006). *Reconciling information exchange and confidentiality a formal approach*. PhD thesis, Rijksuniversiteit Groningen.

van Benthem, J., Gerbrandy, J., Hoshi, T., and Pacuit, E. (2009). Merging frameworks for interaction. *Journal of Philosophical Logic*, 38(5):491–526.

Van Benthem, J., Van Eijck, J., and Kooi, B. (2006). Logics of communication and change. *Information and Computation*, 204(11):1620–1662.

Van Ditmarsch, H., Ghosh, S., Verbrugge, R., and Wang, Y. (2014). Hidden protocols: Modifying our expectations in an evolving world. *Artificial Intelligence*, 208(1):18–40.

Van Ditmarsch, H., Van der Hoek, W., and Kooi, B. (2007). *Dynamic Epistemic Logic*. Springer Science & Business Media.

Van Eijck, J. (2012). Perception and change in update logic. In Van Eijck, J. and Verbrugge, R., editors, *Games, Actions and Social Software*, volume 7010 LNCS, pages 119–140. Springer.

Van Kooten Niekerk, A. and Wijmer, S. (1985). *Verkeerde vriendschap: Lesbisch leven in de jaren 1920-1960*. Feministische Uitgeverij Sara, Amsterdam.

Verbrugge, R. (2009). Logic and social cognition: The facts matter, and so do computational models. *Journal of Philosophical Logic*, 38(6):649–680.

Wang, Y. (2010). *Epistemic modelling and protocol dynamics*. PhD thesis, Universiteit van Amsterdam.

Wijermans, N., Jager, W., Jorna, R., and van Vliet, T. (2008). Modelling the dynamics of goal-driven and situated behaviour. In *The Fifth Conference of the European Social Simulation Association (ESSA 2008)*, pages 1–7, Brescia, Italy.

Wittgenstein, L. (1953). *Philosophical Investigations (Philosophische Untersuchungen) Eng. & Ger.* Oxford. Translated and edited by Anscombe, GEM.

Zoethout, K. and Jager, W. (2009). A conceptual linkage between cognitive architectures and social interaction. *Semiotica*, 2009(175):317–333.