UNIVERSITÉ DU
LUXEMBOURG

University of Luxembourg

Faculty of Science, Technology and Communication

Deontic Agency and Moral Luck

Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of *Master in Information and Computer Sciences*

Supervisor
Prof. Dr. Leon van der Torre

Author
Paul Meder

Reviewer
Prof. Dr. Martin Theobald

Student Number
009060593f

First Advisor
Dr. Xavier Parent

Date
August 2018

Second Advisor
Priv.-Doz. Dr.-Ing. Christoph Benzmüller

Abstract

This work presents temporal STIT I/O logic, an I/O logic based on temporal STIT logic, and documents its investigation with the proof assistant tool Isabelle/HOL. We show how to semantically embedding temporal STIT logic as well as the out_2 operator in HOL. Implementing those embeddings and also the already existing embedding of the out_1 operator into Isabelle/HOL framework, enables the application of higher-order automatic theorem provers for automated reasoning tasks in temporal STIT I/O logic. Finally, we relate our logic to a more philosophical topic called moral luck, identify which aspects of moral luck can be studied by it, and use examples from this subject as test cases for the logic.

Keywords: Deontic logic, Input/Output logic, STIT logic, Higher-order logic, Semantical Embedding, Moral luck

Declaration of Honor

I hereby declare on my honor that I am the sole author of the present thesis. I have conducted all work connected with the thesis on my own.

I only used those resources that are referenced in the work. All formulations and concepts adopted literally or in their essential content from printed, unprinted or Internet sources have been cited according to the rules for academic work and identified by means of footnotes or other precise indications of source.

This thesis has not been presented to any other examination authority. The work is submitted in printed and electronic form.

Luxembourg, August 2018

Paul Meder

Acknowledgments

I want to thank the University of Luxembourg as well as the professors for their lectures during my studies. The courses allowed me to improve my knowledge and skills about computer science and prepared me well for the master thesis.

I wish to express my gratitude to Prof. Dr. Leon van der Torre and Dr. Xavier Parent for allowing me to participate at this project. During our meetings they provided me with relevant advices and were so kind to review my thesis.

Further on, I would like to sincerely thank Priv.-Doz. Dr.-Ing. Christoph Benzmüller who offered me valuable suggestions to realize my work.

Moreover, I am thankful to Mr. Ali Farjami for his help and advices throughout this work.

Finally, I am grateful to all my family members and friends for their support and encouragement during my studies and this project.

Contents

List of Figures

1 | Introduction

1.1 Context

The field of logic that deals with normative concepts such as obligation, prohibition, and permission, is known as Deontic logic. It is a very well studied area of philosophy and mathematical logic. When it comes to the formalization of knowledge about norms, for instance, the representation of legal knowledge, deontic logic is an obvious choice. Various kinds of deontic logics have been developed over the years. We have the *traditional* deontic logic, which includes *Standard Deontic Logic (SDL)*, a modal logic of type KD, and *Dyadic Deontic Logic (DDL)*, which have been proposed to deal with *contrary-to-duty (CTD)* reasoning. The so-called *norm-based* deontic logics represent another family. In contrast to traditional deontic logic, the deontic operators are not evaluated with a possible worlds semantics. For a norm-based deontic logic, the deontic modalities are analyzed with reference to a set of explicitly given norms. Such a framework investigates which norms apply for a given input set, referred to as facts, and a set of explicitly given conditional norms, referred to as normative system. A particular framework that falls within this category, is called *input/output (I/O) logic*. It has been developed by Makinson and van der Torre [23] and it is one of the latest achievements in the area of deontic logic and gained a high recognition in the AI community. Each of those deontic logics is used in different application domains such as legal and ethical reasoning, or normative multi-agent systems. For the latest, one needs to combine deontic logic with modalities of agency. Horty's work [19] focuses on the combination of deontic logic and a modal logic of action, known as STIT theory, and it is considered as a major and important contribution in the domain of deontic agency.

Deontic logic has recently received new attention in computer science with regard to automated reasoning. One of the latest paper [6] by the *Individual and Collective Reasoning (ICR)* group, proposes an infrastructure for the automation of deontic and normative reasoning and discusses the development of computational tools for reasoning based on deontic logic. The presented infrastructure supports a wide variety of different variants of deontic logics. But how did the authors encode deontic logic into a computer system, so that logical reasoning can be simulated on it? They took a particular deontic logic and performed a so-called *shallow semantical embeddings (SSE)* [1] of that logic in *higher-order logic (HOL)*. By using this approach, HOL serves as a meta-logic in which the syntax and semantics of that particular deontic logic can be represented and modeled. The embedded logic was then implemented into a computer software, *Isabelle/HOL* [26]. It is a proof assistant tool which uses HOL as its background logic. A variety of state-of-the-art higher-order automatic theorem provers (ATPs) and model finders are integrated into *Isabelle* and can be called by the framework's supported tools. Basically speaking, an ATP is a computer software which takes a logical statement as its input and automatically generates a proof for it by using a set of axioms, theorems, and hypotheses. Due to the technique of semantical embedding, the authors were able to represent *SDL* [2], *DDL* by Carmo and Jones [5], as well as the out_1 operator for I/O logic [7] in higher-order logic and their embeddings were implemented into frameworks such as *Isabelle/HOL*. This approach allows for applying ATPs in order to solve certain problems modeled by those logics. Those works form more or less the basis of the literature with regard to deontic logic, HOL and computer supported reasoning. However, besides those deontic logics, several non-classical logics such as several modal logics [2], including *SDL*, conditional logic [4], and many others have also been embedded in HOL and verified within the *Isabelle/HOL* framework.

1.2 Research questions

In this thesis, we investigate and contribute to the area of deontic logic, agency and automated reasoning. We are particularly interested in further observing and verifying Input/Output (I/O) logic with the proof assistant tool *Isabelle/HOL* [26], but also explore STIT logic and its combination with I/O logic with the help of the framework. By the earlier mentioned semantical embedding approach, STIT logic and I/O logic can be formulated in such a way that they can be represented in HOL and therefore be simulated within the *Isabelle/HOL* tool. For this thesis, we therefore identified and focus on the following research questions:

RQ1: Due to the recently notable success of embeddings of non-classical logics in HOL and their verifications in *Isabelle/HOL*, is it possible to realize an embedding of STIT logic in HOL or are there any limitations?

RQ2: So far, the literature only documents an embedding of the out_1 operator in HOL [7]. Can the work of I/O logic in HOL be extended by providing a semantical embedding of the out_2 operator?

RQ3: Typically, I/O logic is used with propositional logic. How do the I/O operators out_1 and out_2 perform when using STIT logic as the base logic?

RQ4: It has been suggested that moral luck can be studied using deontic logic, but which aspects can be analyzed and which aspects are out of reach?

1.3 Goals and methodology

We want to investigate a logic which is composed of two logics. I/O logic covers the deontic part whereas STIT logic is used for the agency part. We consider a STIT logic augmented with some temporal operators, known as temporal STIT logic [22], shortly T-STIT logic. For I/O logic, we focus on the out_1 and out_2 operator. The question is how to combine these two logics? In its traditional form, I/O logic uses the language of classical propositional logic as its base logic. To combine them, we change the language of the base logic to a language of T-STIT logic. We will refer to the combined logic as *T-STIT I/O logic*.

In order to answer our research questions, the first goal of this thesis is to come up with a semantical embedding for T-STIT logic in higher-order logic (HOL) and provide an implementation of it in *Isabelle/HOL*. Lorini's T-STIT logic provides us with a possible worlds semantics, which comes in useful when embedding the logic in HOL. In particular, the semantics introduce a so-called temporal Kripke STIT model, a multi-relational Kripke model with specific constraints on each accessibility relation. We will test the implementation by trying to generate proofs for the axioms of T-STIT logic as well as for some laws for STIT logic from Horty's work [19]. *Isabelle/HOL* supports automatic reasoning tools, such as $Sledgehammer$ [11] and $Nitpick$ [10], which can be used to prove or disprove logical statements, respectively.

Next, the focus lies on the semantical embedding for the I/O operators out_1 and out_2 in HOL. Those embeddings will also be implemented into *Isabelle/HOL*. The work in [7] documents the first experiments with it and already presents a semantical embedding of out_1 in HOL. The embedding of out_1 is based on its traditional formulation. However, for the out_2 operator there is a translation into modal logic, which will be used to realize the embedding. Having implemented the embeddings of the two operators and T-STIT logic into *Isasbelle/HOL*, we are able to investigate their combination, which we refer to as T-STIT I/O logic, and apply the operators to a number of different examples in order to verify their correctness.

The final part is to apply T-STIT I/O logic to examples of moral luck. Moral luck describes the phenomenon when an agent is held accountable for his actions and its consequences even though it is clear that the agent was neither in full control of his actions nor its consequences. It conflicts therefore with the ethical principle that agents are not morally responsible for actions that they are unable to control. Horty already observed that his two defined obligation operators have something to say about moral luck [19] (Chapter 5, page 121). In this thesis, we want to go into more detail and see which aspects of moral luck, deontic logic is able to analyze. Therefore we dedicate a section of this thesis with a survey of different kinds of moral luck and we use it as a test case for the logic. In particular, we focus on a specific kind of moral luck, known as resultant moral luck. It is concerned with the consequences of an agent's actions. Due to T-STIT logic, we are able to formalize and model such scenarios and by using I/O logic, we are able to tell what is obligatory in those situations.

By achieving our proposed goals, this thesis contributes the following to the literature of deontic logic, agency and automated reasoning. First, it documents an semantical embedding of T-STIT logic, a logic of action, in HOL and how it has been implemented in *Isabelle/HOL*. Next, we extend the work of I/O logic in HOL by providing an embedding for the out_2 operator. Since we also investigate T-STIT I/O logic, we explore further aspects of I/O logic with a more expressive base logic and give more insight of it with regard to moral luck.

1.4 Interdisciplinary aspects

Testing our logic on examples from a phenomenon such as moral luck, can be extremely valuable and indicate how well the logic can deal with such real-world scenarios and if it is possible to use it outside of theoretical computer science. Such involvements include artificial intelligence (AI) applications where logic is considered as one of the most important and powerful tools used in the development.

Self-driving cars or autonomous weapons are two modern and popular examples of technologies where AI is used as a key concept. In contrast to manual driving cars, driverless cars are considered to be safer, more fuel-efficient, and their popularity is increasing day by day. But putting, for instance, AI behind the wheel of a car gives raise to ethical problems. When the car is faced with moral dilemmas, how should it behave in those lose-lose situations? Who can be held accountable when an autonomous vehicle ends up in a tragic accident? Imagine a scenario with an unavoidable crash, the self-driving car has to make a choice between either saving the life of the driver or the lives of five pedestrians. It comes down to the reasoning power of AI in the driverless car, which now has to decide between life and death. What does the AI consider as the right choice? The most logical approach would be to minimize casualties. In this case, it would mean that killing one person is better than killing five people, thus the AI will favor the lives of pedestrians over the life of the driver. However, considering this as the right approach is highly doubtable. Identifying the right choice is extremely complex in such circumstances, if not impossible, and it is a quite challenging task that computer scientists are facing when developing the logic of such an AI and its implementation. Not only are they concerned with the correctness of the logic but they also have to adjust it in such a way that the AI is able to deal with such ethical decisions. Such cases are illustrating that AI has sparked more ethical debates than any other technology before it.

The rest of the introduction section is used to provide a general overview of STIT theory. We give a general overview of the logic of agency, known as STIT logic, that originates in the domain of philosophy of action. We present the motivations behind this logic and the key concepts of it.

1.5 STIT theory

Agency deals with what agents can bring about and actions are a way to bring about some state of affairs. A particular logic of agency is the logic of STIT which deals with choices and strategies for individual agents as well as for groups of agents. This logic originates from the domains of the philosophy of action and has been proposed in 1990 [9]. The objective was to formulate semantics about agents and what they do. This allows analyzing the needs for a general theory of an agent making a choice among alternatives that lead to an action. The works of Belnap et al. [8] and Horty [19] are considered as major contributions in the area of STIT theory. Over the recent years, people in computer science have also done some work in STIT theory. Most of it includes proof theory and axiomatization [31].

STIT refers to the acronym *seeing to it that* and originally the logic was embedded in a branching time indeterministic framework. The time is represented in the form of a tree and the branches are referred to as *histories*. Any choice made or action performed by an agent restricts the future to a subset of these histories. According to Horty, histories can be thought of as 'possible outcomes of actions' and an action or a choice K of an agent may therefore be represented by a set of outcomes. Further, this theory consists of an *agentive* modality in order to formulate the idea that an agent causes some state of affairs. Usually one writes this modality as $[\alpha\, stit : \varphi]$ and can be interpreted as *agent α sees to it that φ*. Over the years, several different kinds of agency operators have been proposed in STIT theory. Belnap and Perloff introduced the logic for the achievement STIT. Besides the strong support for the theory of the achievement STIT, its logic is rather complex. In particular, Brian Chellas, who established the first semantics of a logic of action in [14], stated the following:

> Belnap and Perloff's "theories of agency are complex, fascinating, and illuminating – without a doubt the most subtle and sophisticated proposals of their kind to date." [15].

The complexity of the achievement STIT might be a reason why Horty stayed away from it when he was studying agency operators that could mix with deontic aspects. Inspired by the work in philosophy of action, Horty introduced the deliberative STIT theories [20] which contain the logic of two operators of agency. The first operator of agency is called *Chellas's STIT* $[\alpha\, cstit : \varphi]$ and it is the most elementary one. It is named after Brian Chellas since it corresponded more or less to his operator $\triangle_\alpha \varphi$ in [14]. In terms of semantics, we have that $[\alpha\, cstit : \varphi]$ holds if and only if agent α makes a choice K and φ is true in all outcomes that could result because of agent α making this choice K. The later one also means that φ can be seen as a necessary consequence of choice K. Besides the Chellas STIT operator, Horty also introduced the *deliberate* stit modality. The operator is typically written as $[\alpha\, dstit : \varphi]$ and can be read as *agent* α *deliberately sees to it that* φ. For an agent α to deliberately see to it that φ, it is required that $[\alpha\, cstit : \varphi]$ and further that there exists at least one outcome in which φ does not hold. So it corresponds to the Chellas's STIT with a negative condition.

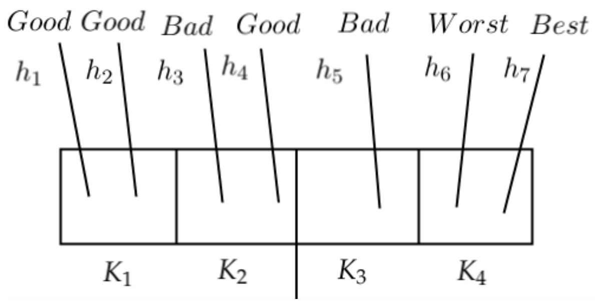


Figure 1.1: Agent facing four choices

Consider Figure 1.1, which will be used as an illustration for some of these concepts. It represents a situation where a single agent α is faced with four choices $K_1, \cdots, K_4$. The seven histories $h_1, \cdots, h_7$ can be thought of as possible outcomes. Besides the choice K_3, each other choice restricts the possible outcomes to two. The labels *Good*, *Bad*, *Best* and *Worst* can be seen as propositions and indicate the result of the outcome. They only hold in the outcomes where they occur. For instance, the proposition *Good* might be interpreted as 'something is about to happen' and holds for the histories h_1, h_2 and h_4. Assuming that agent α is making the choice K_1 then $[\alpha\, cstit : Good]$ is true since for every outcome of the choice K_1, namely h_1 and h_2, the proposition *Good* is true. Moreover, we have that $[\alpha\, dstit : Good]$ holds as well for the choice K_1. Indeed we have that $[\alpha\, cstit : Good]$ and there is an outcome, for instance, h_3, where the proposition

Good does not hold.

STIT logic distinguishes itself from other logics of agency such as Coalition logic or Alternating-time temporal logic. Those logics were designed in order to express what an agent is $able$ to do. This feature is also captured by STIT logic, however, what clearly makes a difference between this logic and the other two, is that STIT logic can additionally express what an agent actually $does$. Consequently, this results in a more expressive power since it allows us to exactly figure out the contributions of an agent to the actual state of affairs or what actually happened. This ability is considered as a major factor in responsibility assignment or distribution. Take for instance the example of a person that has been killed. When it comes to tracking the agents who committed this crime, we don't only want to identify those agents who had been able to perform the act of killing someone but we want to know who is responsible for the death of the person. Or in other words, we precisely want to charge the agent that killed the person. While Coalition logic and Alternating-time temporal logic are only capable of capturing the part of being able or empowered to do, STIT logic accomplishes to single out both parts due to its greater expressive power. This is what makes this logic a suitable candidate when we want to investigate some important differences in the general context of agency. Depending on the context, further expressive power can be achieved by adding operators from different logics such as epistemic or deontic logic. For instance, if we want to express what an agent knowingly does, we have to add epistemic operators whereas deontic operators or even a combined deontic-stit operator are essentially needed when we want to reason about what an agent ought to do.

Moreover, STIT logic is not restricted to individual agents, but it can also express *collective* actions meaning what a group of agents does. Thus it is concerned with a set of agents which is often referred to as a *coalition*. To express such statements, the language of STIT has to be extended with constant symbols which denote then the sets of agents. Those symbols vary from literature to literature. The most common ones are Agt, A or Γ. The group or coalition A sees to it that φ is then stated by $[A\, stit:\ \varphi]$. By doing so, one leaves the field of mono-agency and enters the domain of multi-agency. When it comes to identifying the responsibility of an agent in a multi-agent system, one needs the ability to contrast the agents. In particular, one should be able to tell that agent α, but not agent β, committed a specific crime. However in the case that agent α and β have both committed the same crime, one needs also to single out the responsibilities that are shared among multiple agents. Such expressions can again be covered by STIT logic. Horty [19] and Broerson [12] provided specific versions of STIT logic regarding what group of agents or coalitions do.

1.6 Structure of the thesis

The thesis is composed of three main parts and is structured in the following way: Chapter 2 presents the logical part. In particular, T-STIT I/O logic, a combination of I/O logic and T-STIT logic. We start by recalling the semantics and proof theory of Makinson's and van der Torre's traditional I/O logic and then how we combine it with T-STIT logic. Chapter 3 contains the part about the implementation. It starts off with an introduction to higher-order logic (HOL) and *Isabelle/HOL*. Next, it documents the semantical embedding of temporal STIT logic in HOL and its implementation into *Isabelle/HOL*. Subsequently, it demonstrates the implementation of I/O operators out_1 and out_2 into *Isabelle/HOL*. Chapter 4 focuses on the conceptual part. It presents the principle of *Moral Luck* and the problem that comes with it. We identify which aspects of moral luck can be related to deontic logic and use examples of moral luck as test cases for our logic. The final chapter summarizes and concludes the work and outlines some directions for future work. Appendix A contains a general overview of deontic logic whereas appendix B presents the distinction between ought-to-be and ought-to-do, two important examples from Horty's book [19] and a critical view on STIT theory. Appendix C includes first a section of Lorini's T-STIT logic [22] to cover its syntax, semantics and axioms. Further, it also contains a semantics based on Horty's work [19]. The final appendix D documents how temporal Kripke STIT models are encoded in Isabelle/HOL in order to evaluate some formulas on a specific model.

2 | The logic

This chapter starts with a section to recall the semantics and proof theory of traditional Input/Output logic in order to provide a basic understanding of this logic. The notations are taken from [23] . Then we introduce T-STIT I/O logic, which is an I/O logic using Lorini's T-STIT logic as its base logic.

2.1 Input/Output logic

Input/Output (I/O) logic was initially developed by Makinson and van der Torre, and introduced in [23]. It is considered as one of the new achievements in deontic logic. I/O logic focuses on the reasoning and studying of conditional norms. Typically, a conditional norm expresses what ought to be the case in a certain situation. Just like modal logic, which can be seen as a family of systems K, D, S4, S5 and so on, I/O logic is also a family of logics.

However, they are different in terms of semantics. Modal logic is usually interpreted in possible world semantics, but I/O logic falls within the category of operational or norm-based semantics. Obligations are not explained by some sets of possible worlds among which some are ideal or at least better than others, but they are referenced to a given set of conditional norms. The meaning of the deontic concepts is given in terms of a set of procedures yielding outputs for inputs. The basic mechanism behind these procedures is that of detachment or modus ponens.

2.1.1 Semantics

Let $\mathcal{L}$ denote the set containing all the formulas of propositional logic. $N \subseteq \mathcal{L} \times \mathcal{L}$ is called a normative system and is a set of pairs of formulas. In I/O logic, a pair $(a, x) \in N$ is referred to as a conditional norm or obligation, where a and x are formulas of propositional logic. a is called the body and represents some situation or condition, whereas x is called the head and represents what is obligatory or desirable in that situation. Thus the pair (a, x) is read as 'given a, it is obligatory that x'. For example, whenever you are driving in your car and you are approaching a right light, it is obligatory that you stop. The unconditional obligation of x is denoted by $(\top, x)$, where $\top$ is representing an arbitrary tautology.

Given a set $A \subseteq \mathcal{L}$, serving as an input set, the main construct of I/O logic has the form $x \in out(N, A)$ and is interpreted as follows, 'given a state of affairs A, x (obligation) is in the output under the norms N'. When A is a set, which consists only of one element a, the curly brackets are omitted and one writes $out(N, a)$. Alternatively, $x \in out(N, a)$ can be formulated as $(a, x) \in out(N)$. In [23], Makinson and van der Torre introduced several different I/O operations. In particular, out_1, out_2, out_3 and out_4 are defined as the four standard or traditional output operations of I/O Logic. To understand their semantics, we first need to recall some further notations.

For a given N, $N(A)$ denotes the image N under A and is formally written as $N(A) = \{x \mid (a, x) \in N \text{ for some } a \in A\}$. Intuitively, $N(A)$ can be thought of as the set of all the heads x detached from the pairs (a, x) such that the body a is an element of A. Alternatively, $N(A)$ can be interpreted as the set containing the direct consequences of the normative system N for some input set A.

Remark In most cases, we use N to denote the set of conditional norms. However, any letter can be used to denote such a set. For instance, in the literature of I/O logic, the letter G is also sometimes used to represent such a set. Consequently, we write then $G(A)$ to refer to the image G under A.

$Cn(A)$ represents the set of logical consequences of A and is formally defined as $Cn(A) = \{x \mid A \vDash x\}$, where $\vDash$ is the propositional consequence relation.

Finally, a set $A \subseteq \mathcal{L}$ of formulas is called *consistent* if $A \nvDash \bot$, where $\bot$ is representing an arbitrary contradiction, and *inconsistent* otherwise.

Example 1 Let $N = \{(a, x), (b, y), (a \wedge b, z)\}$ and $A = \{a, b\}$. Then, we have the following:

A	$Cn(A)$	$N(A)$	$N(Cn(A))$
$\{a, b\}$	$Cn(\{a, b\})$	$\{x, y\}$	$\{x, y, z\}$

For some set of facts A and some set of conditional norms N, the output operators out_1, out_2, out_3 and out_4 are formulated as follows:

- $out_1(N, A) = Cn(N(Cn(A)))$
- $out_2(N, A) = \bigcap\{Cn(N(V)) \mid A \subseteq V, V \text{ complete}\}$
- $out_3(N, A) = \bigcap\{Cn(N(B)) \mid A \subseteq B = Cn(B) \supseteq N(B)\}$
- $out_4(N, A) = \bigcap\{Cn(N(V)) \mid A \subseteq V \supseteq N(V), V \text{ complete}\}$

A set of formulas is considered as *complete* if it is either *maximal consistent* or equal to $\mathcal{L}$. For this thesis, we only focused on the operators out_1 and out_2.

The operation out_1 is also called the *simple-minded* output and the basic idea behind it, is the following: first, we are given a set of formulas A, which represents some facts and we close it under logic consequence. As a next step, we pass this closed set to the normative system. The obtained set of formulas represents the obligations. Finally, those obligations are closed again under logical consequence.

Example 2 Let $N = \{(a, x), (a \vee b, y), (b, z)\}$ and $A = \{a\}$.

A	$Cn(A)$	$N(Cn(a))$	$out_1(N, A)$
a	$Cn(a)$	$\{x, y\}$	$Cn(\{x, y\})$

We have that $x \in out_1(N, A)$ as well as $y \in out_1(N, A)$ since both x and y are obviously in $Cn(\{x, y\})$. Also, we have that $(x \wedge y) \in out_1(N, A)$ since $(x \wedge y) \in Cn(\{x, y\})$.

However, the *simple-minded* output might not be enough. For instance, out_1 fails when it comes to reasoning by cases. Suppose that a set of conditional norms N contains the pairs (a, x) and (b, x). When $a \vee b$ is implied by some facts, then we should be able to conclude that x is the case. Formally, for $N = \{(a, x), (b, x)\}$ and $A = \{a \vee b\}$, we get that $out_1(N, A) = Cn(\varnothing)$, where $Cn(\varnothing)$ denotes the set containing all possible tautologies. Therefore we don't have that $x \in out_1(N, a \vee b)$ since $a \vee b$ is not a tautology. In order to support such reasoning by cases, we use the operation out_2, which is also referred to as *basic* output. To show that $x \in out_2(N, A)$, we need to consider the complete set V. There are two choices, either V equals to $\mathcal{L}$ or V is an maximal consistent extension (MCE) of $a \vee b$. In the first case, we get that $Cn(N(V)) = Cn(x)$. In the second case, by maximal consistency of V, we have that either a or b are in V. If $a \in V$ then $Cn(N(V)) = Cn(x)$. Identically for $b \in V$. Thus $out_2(N, A) = Cn(x) \cap Cn(x)$ and $x \in out_2(N, A)$.

In [23], it has been shown that out_2 can be formalized in terms of modal logic. In particular, we have: $x \in out_2(N, A)$ if and only if $x \in Cn(N(\mathcal{L}))$ and $N^{\Box} \cup A \vdash_S \Box x$ for any modal logic **S** with $\mathbf{K_0} \subseteq \mathbf{S} \subseteq \mathbf{K_{45}}$. The notation $N^{\Box}$ denotes the set of all modal formulas of the form $b \to \Box y$, such that $(b, y) \in N$. We have that $N^{\Box} \cup A \vdash_S \Box x$ if for all the elements $y_i \in Y$, such that Y is a finite subset of $N^{\Box} \cup A$, it holds that $(\bigwedge y_i \to \Box x) \in \mathbf{S}$. Alternatively, it means that $\bigwedge y_i \to \Box x$ is a valid formula in **S**.

Example 3 Let $N = \{(a, x), (b, x)\}$ and $A = \{a \vee b\}$.

In order to check if $x \in out_2(N, A)$, we first need to check that $x \in Cn(N(\mathcal{L}))$. This is straightforward, since $Cn(N(\mathcal{L})) = Cn(\{x\})$, and obviously we have that $x \in Cn(\{x\})$.

Next, we have $N^{\Box} \cup A = \{a \to \Box x, b \to \Box x, a \vee b\}$. The formula $((a \to \Box x) \wedge (b \to \Box x) \wedge (a \vee b)) \to \Box x$ is valid for any modal logic **S** with $\mathbf{K_0} \subseteq \mathbf{S} \subseteq \mathbf{K_{45}}$. Thus $x \in out_2(N, A)$.

2.1.2 Proof Theory

In terms of proof theory, I/O logics are characterized by derivation rules about norms. Given a set of norms N, a derivation system is the smallest set of norms which extends N and is closed under certain derivation rules.

- (SI) Strengthening the input: from (a, x) to (b, x) whenever we have $\vDash b \to a$
- (WO) Weakening the output: from (a, x) to (a, y) whenever we have $\vDash x \to y$
- (AND) Conjunction of the output: from (a, x) and (a, y) to $(a, x \wedge y)$
- (OR) Disjunction of the input: from (a, x) and (b, x) to $(a \vee b, x)$
- (CT) Cumulative transitivity: from (a, x) and $(a \wedge x, y)$ to (a, y)

$deriv_1$ denotes the derivation system for the I/O logic operator out_1 and is formed by the rules SI, WO and AND. Adding OR to $deriv_1$ gives $deriv_2$, whereas extending $deriv_1$ with CT gives $deriv_3$. They represent the derivation systems for the output operators out_2 and out_3, respectively. The derivation system for out_4 is called $deriv_4$ and it is closed under all of the five rules.

Example 4 Let $N = \{(a \vee b, x)\}$. Then we have $(b, x \vee y) \in deriv_1(N)$.

1.	$(a \vee b, x)$	By assumption
2.	(b, x)	1, SI
3.	$(b, x \vee y)$	2, WO

Soundness and completeness hold for each of the four derivation systems $deriv_i$ with respect to the corresponding output operator out_i for $i \in \{1 \cdots 4\}$. See [23]. By soundness, we have that $deriv_i(N) \subseteq out_i(N)$ and by completeness, we have that $out_i(N) \subseteq deriv_i(N)$. This means that we get the following theorem:

Theorem 2.1.1 For a given $N \subseteq \mathcal{L} \times \mathcal{L}$, we have that $\psi \in out_i(N, \varphi)$ if and only if $(\varphi, \psi) \in deriv_i(N)$.

From the previous example, we get that $(b, x \vee y) \in deriv_1(N)$. It can easily be verified that $(x \vee y) \in out_1(N, b)$. Indeed, we have that $out_1(N, b) = Cn(\{x\})$ and $(x \vee y) \in Cn(\{x\})$.

2.2 T-STIT I/O logic

Since I/O logic only uses propositional logic as its base logic, it is limited in its expressive power. For instance, it is not able to capture expressions dealing with concepts such as agent and action. Those concepts are a crucial part of agent theory and multi-agent systems, and STIT logic is able to express notions such as agent and action. Thus, one way to increase the expressiveness of I/O logic is to a particular STIT logic as the basis. Xin Sun [29] used the individual deliberative STIT operator as the basis to formulate I/O STIT logic and proved the completeness for it. Additionally, he showed that it is free from Ross's paradox.

In our case, to combine I/O logic with T-STIT logic, we will no longer consider $\mathcal{L}$ as the base logic for the I/O operations, but instead we will use $\mathcal{L}_{\text{T-STIT}}$ which denotes the language of T-STIT logic. The appendix contains a detailed section which covers the syntax, semantics and axioms of Lorini's T-STIT logic [22]. We will briefly recall the language's syntax and give an interpretation to its formulas.

Given a set $\mathbb{P}$ of propositional letters and a finite set of agents Agt. For every $p \in \mathbb{P}$ and $\alpha \in Agt$, the language of $\mathcal{L}_{\text{T-STIT}}$ is defined by the following BNF (Backus Normal Form):

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box\varphi \mid [\alpha]\varphi \mid [Agt]\varphi \mid G\varphi \mid H\varphi$$

Intuitively, $\Box\varphi$ is read as "φ is necessarily true". Operators of the form $[\alpha]$ denote the Chellas's STIT operator and can be read as "agent α sees to it that φ". $[Agt]$ represents the group STIT operator and can be read as "all the agents see to it that φ by acting together". The operators G and H denote the strict future and strict past operators, respectively. $G\varphi$ is read as "φ will always be true in the future" whereas $H\varphi$ is read as "φ has always been true in the past".

For T-STIT I/O logic, we have then that $N \subseteq \mathcal{L}_{\text{T-STIT}} \times \mathcal{L}_{\text{T-STIT}}$ and $A \subseteq \mathcal{L}_{\text{T-STIT}}$ and our logic can use formulas from T-STIT logic. For instance, a conditional norm can now contain formulas such as $[\alpha]a$ where α denotes an agent and a a propositional letter. The semantics of the operator out_1 still remains the same and also the derivation system $deriv_1$ is still composed of the rules SI, WO and AND.

Example 5 Let α and β represent two agents and a, b, c and d be propositional letters. Consider $N = \{(a, [\alpha]c), (a, [\beta]d)\}$ and $A = \{a\}$. Then, we have:

A	$Cn(A)$	$N(Cn(A))$	$out_1(N, A)$
a	$Cn(a)$	$\{[\alpha]c, [\beta]d\}$	$Cn(\{[\alpha]c, [\beta]d\})$

Example 6 Let α and β represent two agents and a, b, c and d be propositional letters. Consider $N = \{(a \vee b, [\alpha]c)\}$, then we have $([\beta]b, [\alpha](c \vee d)) \in deriv_1(N)$. Indeed we get that:

1.	$(a \vee b, [\alpha]c)$	By assumption
2.	$([\beta]b, [\alpha]c)$	1, SI
3.	$([\beta]b, [\alpha](c \vee d))$	2, WO

We also preserve the same semantics for the operator out_2 as well as the derivation system $deriv_2$. However, in order to semantically embed out_2 in HOL, we don't consider its traditional formulation but we will use its translation into modal logic.

3 | The implementation

This chapter presents the implementation of T-STIT logic as well as I/O logic into *Isabelle/HOL*. We start with a section about higher-order logic (HOL), the logic used by *Isabelle/HOL*. This is necessary to understand the embedding of T-STIT logic in HOL and the implementation of it in *Isabelle/HOL*, covered in the following sections. Then we discuss the limitations of the implementation. In appendix D, we further show to formulate specific temporal Kripke STIT models in *Isabelle/HOL*. The next part deals with the implementation of I/O logic in *Isabelle/HOL* with the focus on the operators out_1 and out_2. The operators are tested on examples with formulas of propositional logic as well as of temporal STIT logic.

3.1 Higher-order logic

Higher order logic (HOL) was formalized by Bertrand Russell [28] and has been introduced in 1908 in order to provide a formal basis for mathematical reasoning. HOL is also known as type theory and got adapted over the years so that in its modern form, it is based on the simply typed λ-calculus, which originates from Alonzo Church's *simple type theory (STT)* [16], and the formulations by Leon Henkin [17]. Type theory became an integral and indispensable part in every subject that deals with computation and logical reasoning, which made it an expressive foundation in the domains of mathematics and computer science.

In HOL, the set of types is denoted by T, whose elements are freely generated from the set of the basic types $\{o, \mu\}$. The former one represents the type of booleans whereas the last one represents the type of individuals. Further whenever we have $\alpha, \beta \in T$ then $\alpha \rightarrow \beta \in T$. $\alpha \rightarrow \beta$ denotes the function type and it is referring to the type of function which takes an input of type α and produces an output of type β. Those function types are constructed by using the symbol $\rightarrow$. So any type τ is generated by the following grammar:

$$\tau ::= o \mid \mu \mid \tau \rightarrow \tau$$

A formula or term in HOL is generated by the following BNF:

$$s, t ::= p_\alpha \mid X_\alpha \mid (\lambda X_\alpha . s_\beta)_{\alpha\rightarrow\beta} \mid (s_{\alpha\rightarrow\beta} t_\alpha)_\beta \mid (\neg_{o\rightarrow o} s_o)_o \mid ((\vee_{o\rightarrow o\rightarrow o} s_o) t_o)_o \mid (\forall_{(\alpha\rightarrow o)\rightarrow o}(\lambda X_\alpha . s_o))_o$$

with $\alpha, \beta \in T$. In the most simple case, terms can be constants of type α, denoted by p_α or variables of type α, denoted by X_α. Further those X_α have to be distinct from p_α. A $constant$ always refers to the same specific object like individual, predicate or connective. A $variable$ can denote any object but does not represent a particular object unless it is stated otherwise.

More complex typed terms can be constructed by using abstraction $(s_{\alpha\to\beta}t_\alpha)_\beta$ and application $(s_{\alpha\to\beta}t_\alpha)_\beta$. *Abstraction* means that we can form a term of type $\alpha \to \beta$ which maps the variable X of type α to a term s of type β. The resulting term can be interpreted as a function. *Application* means that we can form a term of type β by applying the term s of type $\alpha \to \beta$, which can be seen as a function, to the term t of type α, which is then the argument. So the resulting term can be interpreted as an outcome of a function to an argument.

Finally, a term is called a formula whenever the term is of type o. Such formulas can represent the logical connectives $\neg_{o\to o}$, $\vee_{o\to o\to o}$ and $\forall_{(\alpha\to o)\to o}$. The symbols $\neg_{o\to o}$ and $\vee_{o\to o\to o}$ denote the *negation* operator and the *disjunction*, respectively, and they work exactly the same ways as in propositional logic. The other symbol $\forall_{(\alpha\to o)\to o}$ represents the *universal quantifier* or *for all quantifier*. The term s of type o gets applied for each variable X of type α. Only if every application is evaluated to true then the *for all quantifier* will evaluate to true. Otherwise it evaluates to false. By using those connectives, other common logical connectives such as $\wedge_{o\to o\to o}, \to_{o\to o\to o}, \equiv_{o\to o\to o}, \bot_o, \top_o$ and $\exists_{(\alpha\to o)\to o}$ can be defined.

3.2 Isabelle/HOL Framework

Isabelle is a generic interactive theorem prover, which has been used to formalize various theorems in mathematics and computer science. It supports a variety of logical frameworks including HOL. *Isabelle/HOL* is the specialization of *Isabelle* for HOL. It is the most widely used instance of Isabelle and can be obtained for free.

The main purpose in *Isabelle/HOL* is to create theories. Embedded in a theory are the types, terms, and formulas of HOL. The type system of HOL is similar to the one of functional programming. For instance, it provides base types, such as *bool*, the type of truth values, and *nat*, the type of natural numbers, and function types which are denoted by the symbol $\Rightarrow$. But, it also allows user-defined types.

The notation $t :: \tau$ expresses that t is a well-type term of type τ. The most obvious form of a term would be a constant but also, just like in functional programming, terms can be formed by applying functions to arguments. Let $\tau_1 \Rightarrow \tau_2$ denote the type of a function f, the argument t is then a term of type

τ_1 and when applied to the function f, $f\,t$ is a term of type τ_2. Further, λ-abstractions may also occur in terms. For instance, $\lambda x.\,x+1$ denotes the function that takes x as an argument and returns $x+1$ as a result.

λ-calculus is a very important concept in Isabelle/HOL. It allows for the replacement of formal by actual parameters. The substitution or computation rule of λ-calculus is defined as:

$$(\lambda x.\,t)\,u = t[u/x]$$

where $t[u/x]$ denotes "t with u substituted for x". For example, the expression $(\lambda x.\,x+1)\,3$ would be transformed into $3+1$. The step from $(\lambda x.\,t)\,u$ to $t[u/x]$ is commonly known as β-reduction and is automatically performed by Isabelle.

A formula is term of type $bool$. For instance, the constants $True$ and $False$ as well as the logical connectives $\neg$, $\wedge$, $\vee$ and $\longrightarrow$ can all been seen as formulas.

Moreover, *Isabelle/HOL* supports very useful reasoning tools such as $Sledgehammer$ [11], which applies automatic theorem provers (ATPs) and satisfiability-modulo-theories (SMT) solver on a given goal, and $Nitpick$ [10], a counter-model generator.

An introduction to *Isabelle/HOL* as well as an in-depth tutorial for functional programming in HOL can be found in the following documentation [26].

3.3 Semantical Embedding

Since HOL is the underlying or background logic of *Isabelle/HOL*, we first need to represent T-STIT logic in HOL. In other words, we have to find equivalent notations of T-STIT logic propositions in HOL. This can be achieved by translating the essential parts of the semantics of T-STIT logic into HOL. This procedure is called the *semantical embedding* of a logic. This allows for using the background logic as a meta-logic in order to validate the semantic truth of syntactic statements in the embedded logic.

Embeddings of a variety of different modal logics in HOL have already been realized [1, 2, 3]. In contrast to HOL, the truth of a propositional formula in a modal logic depends on its context; the possible world where it is evaluated. Thus in order to embed a modal logic in HOL, the principle of a possible world semantics, also known as Kripke semanitcs, needs to be maintained. Since Lorini provided a Kripke semantics for T-STIT logic [22], the techniques from these works can be applied for our embedding.

By introducing a new type i to denote a possible world, the propositions of T-STIT logic are mapped to associated HOL terms of type $i \rightarrow o$, which will be abbreviated as σ. By doing so, the propositional formulas can be represented as functions from possible worlds to truth values in HOL, which allows for explicitly evaluating the truth of a formula in a particular world. For each type α of T-STIT logic, the embedding of α, denoted by $\lceil \alpha \rceil$, is defined as:

$$\lceil \mu \rceil = \mu$$

$$\lceil o \rceil = \sigma = i \to o$$

$$\lceil \alpha \to \beta \rceil = \lceil \alpha \rceil \to \lceil \beta \rceil$$

In HOL, the connectives such as $\neg$ and $\vee$ are of type $o \to o$ resp. $o \to o \to o$. The negation $\neg$ takes one boolean input and returns a boolean output whereas the $\vee$ connective takes two boolean inputs and produces an boolean output. However, due to the lifted propositions of T-STIT logic, the type of the connectives of that logic also have to be mapped. Meaning that for instance $\neg$ and $\vee$ are of type $\sigma \to \sigma$ resp. $\sigma \to \sigma \to \sigma$. To distinguish them from the connectives of HOL, we highlight the lifted connectives in blue. Those lifted connectives are $\neg$, $\vee$, $\wedge$, $\to$ and $\equiv$ and abbreviate the following HOL terms:

$$\neg_{\sigma \to \sigma} := \lambda \varphi_\sigma . \lambda w_i . \neg(\varphi w)$$

$$\vee_{\sigma \to \sigma \to \sigma} := \lambda \varphi_\sigma . \lambda \psi_\sigma . \lambda w_i . (\varphi w \vee \psi w)$$

$$\wedge_{\sigma \to \sigma \to \sigma} := \lambda \varphi_\sigma . \lambda \psi_\sigma . \lambda w_i . (\varphi w \wedge \psi w)$$

$$\to_{\sigma \to \sigma \to \sigma} := \lambda \varphi_\sigma . \lambda \psi_\sigma . \lambda w_i . (\varphi w \to \psi w)$$

$$\equiv_{\sigma \to \sigma \to \sigma} := \lambda \varphi_\sigma . \lambda \psi_\sigma . \lambda w_i . (\varphi w \leftrightarrow \psi w)$$

The lifted negation operator takes a term φ of type σ and a world w of type i. Then it returns the negation of the term φ at the world w. The lifted disjunction takes two terms φ and ψ of type σ and a world w of type i. It returns the disjunction of both terms at that world w. The other lifted connectives work similar to the lifted disjunction.

In modal logic, the $\Box$ operator is evaluated with respect to an accessibility relation R. To tell if two possible worlds are related or not, a constant symbol r of type $i \to i \to \sigma$ is introduced and associated with the accessibility relation R of modal logic. A relation can be seen as a ternary function which takes two possible worlds as its input and returns a boolean output which tells if the two possible worlds are accessible via the relation or not. This allows us then to represent the $\Box$ operator as a quantification over all possible worlds, satisfying an accessibility relation. The same technique can be used for the $\Box$ operator in T-STIT logic, the group stit operator $[Agt]$ as well as for both temporal operators G and H. Thus, those type-raised HOL connectives are defined as follows (where $r^{\Box}$, r^{Agt}, r^{G} and r^{H} denote the corresponding accessibility relations):

$$\Box_{\sigma \to \sigma} := \lambda \varphi_{\sigma}.\lambda w_i.\forall v_i.(r^{\Box}_{i \to i \to \sigma} wv) \to \varphi v$$

$$[Agt]_{\sigma \to \sigma} := \lambda \varphi_{\sigma}.\lambda w_i.\forall v_i.(r^{Agt}_{i \to i \to \sigma} wv) \to \varphi v$$

$$G_{\sigma \to \sigma} := \lambda \varphi_{\sigma}.\lambda w_i.\forall v_i.(r^{G}_{i \to i \to \sigma} wv) \to \varphi v$$

$$H_{\sigma \to \sigma} := \lambda \varphi_{\sigma}.\lambda w_i.\forall v_i.(r^{H}_{i \to i \to \sigma} wv) \to \varphi v$$

The lifted $\Box$ operator takes a term φ of type σ and a world w of type i and then checks for all worlds v of type i if the worlds w and v are related then the term φ is true at the world v. The lifted $[Agt]$, G and H operators work in the same way, using their corresponding relations.

To evaluate the individual STIT operator $[i]$, the corresponding accessibility relation R_i of the agent i has to be considered. Thus this operator is parametric to its relation and has to be defined differently than the other ones.

$$[i]_{(i \to i \to o) \to \sigma \to \sigma} := \lambda r_{i \to i \to \sigma}.\lambda \varphi_{\sigma}.\lambda w_i.\forall v_i.(rwv) \to \varphi v$$

Finally, we need to encode the notation of validity for T-STIT logic propositions (denoted by $\vDash^{T\text{-}STIT}$). This allows for grounding the lifted terms of type σ to type boolean o.

$$valid_{\sigma \to o} := \lambda \varphi_{\sigma}.\forall w_i.\varphi w$$

The operator $valid$ takes a term φ of type σ and returns true if that term is true when applied to any world w of type i. In the remainder, the notation $\lfloor \cdot \rfloor$ will be used as an appropriate abbreviation for validity.

3.4 Implementation

The implementation of the above described semantical embedding of T-STIT logic in HOL, into *Isabelle/HOL* will be further illustrated in this section.

3.4.1 Primitives

The following primitive types have been defined and used as a basis for the embedding:

- Type i denotes the type of possible worlds for Kripke Semantics.
- Type $bool$ denotes the type of meta-logical truth values (i.e. $True$ and $False$) and is already provided by *Isabelle/HOL*.

Based on those, the following function types have been defined:

- Type σ denotes the set of all functions of type $i \Rightarrow bool$ and represents propositions in the embedded logic. Those propositions are evaluated in particular worlds.
- Tye α denotes the set of all function of type $i \Rightarrow i \Rightarrow bool$ and represents the individual relation R_i for possible agents.

Further, the following constants have been introduced:

- The constant aw of type i represents the designated actual world.
- The constants $a1$ and $a2$ of type α represent the equivalence relations R_1 and R_2 for agent 1 respectively agent 2.
- The constant r_box of type $i \Rightarrow i \Rightarrow bool$ represents the equivalence relation $R_{\Box}$.
- The constant r_agt of type $i \Rightarrow i \Rightarrow bool$ represents the equivalence relation R_{Agt}.
- The constant r_G of type $i \Rightarrow i \Rightarrow bool$ represents the binary relation R_G.
- The constant r_H of type $i \Rightarrow i \Rightarrow bool$ represents R_H, the inverse relation of R_G.

3.4.2 Connectives

The basic connectives are lifted to type σ and are defined as follows:

- $\neg\varphi \equiv \lambda w.\ \neg\varphi(w)$
- $\varphi\wedge\psi \equiv \lambda w.\ \varphi(w) \wedge \psi(w)$
- $\varphi\vee\psi \equiv \lambda w.\ \varphi(w) \vee \psi(w)$
- $\varphi\rightarrow\psi \equiv \lambda w.\ \varphi(w) \longrightarrow \psi(w)$
- $\varphi\equiv\psi \equiv \lambda w.\ \varphi(w) \longleftrightarrow \psi(w)$

The operator $\Box$ represents necessity with respect to the relation $R_\Box$. The formula $\Box\varphi$ can be interpreted as "φ is true regardless what every other agent does" and is defined as follows:

– $\Box\varphi \equiv \lambda w.\ \forall v.\ (w\, rbox\, v) \longrightarrow \varphi(v)$

For a given world w, the proposition $\Box\varphi$ is evaluated to $True$ if for all the worlds v such that they are related via $R_\Box$ to w (i.e. those worlds are alternatives to w), v holds φ.

The operator $\Diamond$ represents possibility with respect to the relation $R_\Box$ and can be defined in two different ways:

– $\Diamond\varphi \equiv \lambda w.\ \exists v.\ (w\, rbox\, v) \longrightarrow \varphi(v)$

or

– $\Diamond\varphi \equiv \neg\Box(\neg\varphi)$

The first statement expresses $\Diamond$ by using the quantifier $\exists$. So, for a given world w, the proposition $\Diamond\varphi$ is evaluated to $True$ if there exists a world v such that v is related to w via $R_\Box$ and v holds φ. In the second one, it is defined as the dual of $\Box$.

3.4.3 STIT operators

In T-STIT logic, Chellas's STIT operator [15] is used as primitive operator of agency. Depending on the literature, the Chellas operator is formulated as $[\alpha\, cstit :]$ or $[\alpha]$.

The statement "agent α sees to it that φ regardless what the other agents do" (or shorten "agent α sees to it that φ), for some agent α in the group of agents, can be stated by the formula $[\alpha\, cstit :\ \varphi]$ or $[\alpha]\varphi$.

Remark In chapter 2, we used the notation $[\alpha]$ to denote the Chellas's STIT operator. However for the implementation, we decided to use the other notation $[\alpha\, cstit :]$.

At every world w, each agent i has a set of worlds that are alternatives to w. Those sets are induced by the agent's relation R_i and can be seen as choices available to agent i.

For each agent i, there exists an individual accessibility relation R_i, which is used in order to evaluate the truth of a $cstit$-operation. So the Chellas operator has to be indexed by a parameter r, in order to refer to the agent who is performing the action.

– $r\, cstit\, \varphi \equiv \lambda w.\ \forall v.\ (r\, w\, v) \longrightarrow \varphi(v)$

The deliberative STIT operator can be defined by using Chellas's STIT operator together with the $\Box$-operator.

– $r\, dstit\, \varphi \equiv (r\, cstit\, \varphi) \wedge \neg \Box \varphi$

The group STIT operator, to capture the fact that φ is guaranteed by an action of all agents, can be expressed with the relation R_{Agt}.

– $gstit\, \varphi \equiv \lambda w.\ \forall v.\ (w\, ragt\, v) \longrightarrow \varphi(v)$

Lastly, the dual of group STIT operator, stating that *the group of agents do not prevent* φ, is expressed by:

– $dualgstit \varphi \equiv \neg gstit(\neg \varphi)$

3.4.4 Temporal Operators

Finally, we have the temporal or tense operators. The G and H operators are used to express facts that are true in the strict future and in the strict past, respectively. They can be defined as follows by using the relation R_G and R_H, respectively:

– $G \varphi \equiv \lambda w.\ \forall v.\ (w\, rG\, v) \longrightarrow \varphi(v)$

– $H \varphi \equiv \lambda w.\ \forall v.\ (w\, rH\, v) \longrightarrow \varphi(v)$

Their dual operators F, respectively P, can be expressed as:

– $F \varphi \equiv \neg G(\neg \varphi)$

– $P \varphi \equiv \neg H(\neg \varphi)$

Other useful temporal operators such as G^* resp. F^*, which in contrast to G resp. F, do also include the present.

– $G^* \varphi \equiv \varphi \wedge (G \varphi)$

– $F^* \varphi \equiv \neg G^*(\neg \varphi)$

3.4.5 Relational Properties

Temporal STIT logic imposes several properties on each accessibility relation. First, we start by declaring definitions for those properties for an arbitrary relation r:

- $reflexive\, r \equiv \forall x.\, (r\, x\, x)$
- $symmetric\, r \equiv \forall x\, y.\, (r\, x\, y) \longrightarrow (r\, y\, x)$
- $transitive\, r \equiv \forall x\, y\, z.\, ((r\, x\, y) \wedge (r\, y\, z)) \longrightarrow (r\, x\, z)$
- $serial\, r \equiv \forall w.\, \exists v.\, (r\, w\, v)$

The relations $R_\Box$, every R_i and R_{Agt} are equivalence relations, meaning that they are $reflexive$, $symmetric$ and $transitive$ at the same time. For instance, the *Isabelle/HOL* constant r_box represent the relation $R_\Box$. To specify r_box as an equivalence relation, we use the earlier mentioned definitions and declare it as an axiomatization in *Isabelle/HOL*:

- axiomatization where
  ax_refl_rbox : "reflexive r_box" and
  ax_sym_rbox : "symmetric r_box" and
  ax_trans_rbox : "transitive r_rbox"

R_{Agt} and every R_i can be defined as equivalence relation in a similar way. Next, we have to impose some properties on the binary relations R_G and R_H. The constant r_G represent the relation R_G, which is declared as transitive and serial, and thus we have the following axiomatization in *Isabelle/HOL*:

- axiomatization where
  ax_trans_rG : "transitive r_G " and
  ax_ser_rG : "serial r_G"

Since R_H is the inverse relation of R_G, we first start by formulating an inverse relation and then specify R_H as the inverse of R_G.

- $inverse\, r\, s \equiv \forall w.\, \forall v.\, (r\, w\, v) \longleftrightarrow (s\, v\, w)$
- axiomatization where
  ax_inverse_rG_rH : "inverse r_G r_H"

3.4.6 Relational Constraints

The framework for temporal STIT logic imposes special constraints on the accessibility relations. Several constraints are making use of the inclusion w.r.t. relations as well as the composition between two binary relations. Those can be defined as follows for two arbitrary relation r_1 and r_2:

– $r_1 \subseteq r_2 \equiv \forall w.\ \forall v.\ (r_1\, w\, v) \longrightarrow (r_2\, w\, v)$

– $r_1 \circ r_2 \equiv \lambda w.\ \lambda v.\ \exists u.\ (r_1\, w\, u) \wedge (r_2\, u\, v)$

By the first constraint $C1$, every R_i has to be included in $R_\Box$. Due to that, we first introduce a general formulation for that constraint, which can then be used for any specific R_i relation.

– $axC1\, r \equiv r \subseteq r\_box$

The constants $a1$ and $a2$ represent the relations R_1 and R_2 for the respective agent. On each relation, we have to impose the constraint $C1$.

– axiomatization where
  axC1_a1 : "axC1 a1" and
  axC1_a2 : "axC1 a2"

Constraint $C2$ expresses the *independence of agents* and it can be implemented in a similar way as $C1$. We start off by defining $C2$ and then impose the constraint on the constants $a1$ and $a2$, which represent relations for the respective agents.

– $axC2\, r\, s \equiv \forall w.\ \forall v.\ ((w\, rbox\, w) \wedge (v\, rbox\, v) \wedge (w\, rbox\, v) \wedge (v\, rbox\, w))$
  $\longrightarrow (\exists x.\ (r\, w\, x) \wedge (s\, v\, x))$

– axiomatization where
  axC2_a1_a2 : "axC2 a1 a2"

Constraint $C3$ says that the choice of a group corresponds to the intersection of the choices of the individual agents in the group. Since the group consists of 2 agents, we can express this constraint as follows:

– $axC3\, r\, s \equiv \forall w.\ \forall v.\ (w\, ragt\, v) = ((r\, w\, v) \wedge (s\, w\, v))$

– axiomatization where
  axC3_a1_a2 : "axC3 a1 a2"

Remark The constraints $C2$ and $C3$ are designed for a framework with only two agents. If one decides an other agent, chances have to done to those definitions accordingly.

The constraints $C4$ and $C5$ for the temporal relations R_G and R_H, make sure that the present is connected to the future and to the past, respectively. They can directly be formulated as:

- axiomatization where
  axC4 : "$\forall w.\ \forall v.\ \forall u.\ ((w\, rG\, v) \wedge (w\, rG\, u)) \longrightarrow ((u\, rG\, v) \vee (v\, rG\, u) \vee (u = v))$" and
  axC5 : "$\forall w.\ \forall v.\ \forall u.\ ((w\, rH\, v) \wedge (w\, rH\, u)) \longrightarrow ((u\, rH\, v) \vee (v\, rH\, u) \vee (u = v))$"

The property of *no choice between undivided histories* is expressed by constraint $C6$ and can be stated by using the definitions of inclusion w.r.t. relation and the composition between two relations.

- axiomatization where
  axC6 : "$r\_G \circ r\_box \subseteq r\_agt \circ r\_G$"

Finally, the last constraint $C7$ can again be directly formulated as:

- axiomatization where
  axC7 : "$\forall w.\ \forall v.\ (w\, rbox\, v) \longrightarrow \neg(w\, rG\, v)$"

3.4.7 Validity

Semantic validity is defined as follows:

- $\lfloor\varphi\rfloor \equiv \forall w.\ \varphi(w)$
- $\lfloor\varphi\rfloor_l \equiv \varphi(aw)$

The first statement asserts that a propositional formula φ is semantically valid if it evaluates to $True$ for all the possible worlds whereas the second one expresses that a propositional formula φ is semantically valid for an actual world if it evaluates to $True$ for the actual world aw.

3.4.8 Axioms - Horty

Although Horty does not explicitly treat proof theory in his book [19], he still does provide axioms for the individual STIT operator and uses them for illustration purposes. Due to the encoding of our embedding in *Isabelle/HOL*, it's pretty straight forward to formulate those axioms and prove them by using the $Sledgehammer$ tool or disprove some statements with counter models generated by $Nitpick$. For instance, Horty mentions the following principles for the $cstit$ operator ([19], Chapter 2, page 17):

RE. $\varphi \equiv \psi \,/\, [a\ cstit : \varphi] \equiv [a\ cstit : \psi]$

N. $[a\ cstit : \top]$

M. $[a\ cstit : \varphi \wedge \psi] \rightarrow ([a\ cstit : \varphi] \wedge [a\ cstit : \psi])$

C. $([a\ cstit : \varphi] \wedge [a\ cstit : \psi]) \rightarrow [a\ cstit : \varphi \wedge \psi]$

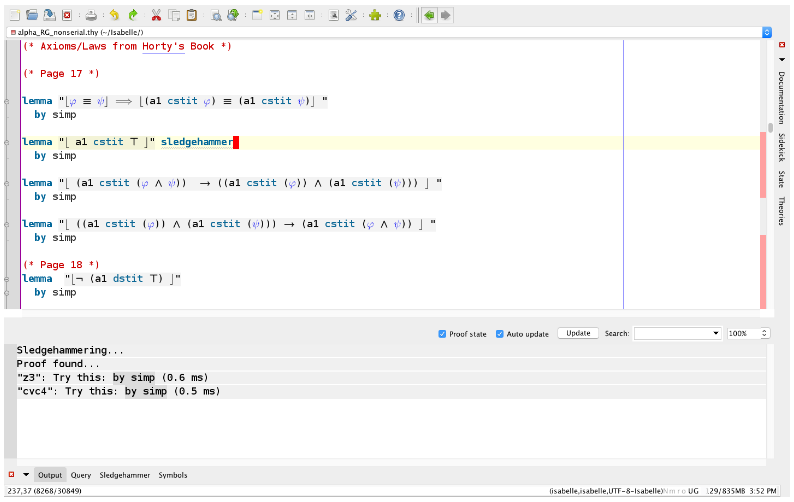
Below the corresponding formulation of those axioms in *Isabelle/HOL*, which can all be proven by $Sledgehammer$.

- $\lfloor \varphi \equiv \psi \rfloor \Longrightarrow \lfloor (a1\ cstit\ \varphi) \equiv (a1\ cstit\ \psi) \rfloor$
- $\lfloor (a1\ cstit\ \top) \rfloor$
- $\lfloor (a1\ cstit\ (\varphi \wedge \psi)) \rightarrow ((a1\ cstit\ \varphi) \wedge (a1\ cstit\ \psi)) \rfloor$
- $\lfloor ((a1\ cstit\ \varphi) \wedge (a1\ cstit\ \psi)) \rightarrow (a1\ cstit\ (\varphi \wedge \psi)) \rfloor$

In *Isabelle/HOL*, we declare those statements as lemmas which allows us to apply $Sledgehammer$ tool in order to prove those. To apply $Sledgehammer$ to one of those lemmas, the only thing one needs to do is to type the command "sledgehammer" next to the lemma. The tool will then call the ATPs and SMT solvers to provide proofs for the statement. Depending on the complexity of the statement, it can take a while until a proof is generated. But as soon as $Sledgehammer$ was successful, the proof appears in the output console of *Isabelle/HOL*. See 3.1. In this case, $Sledgehammer$ found two proves, which were provided by two supported SMT solvers, "z3" and "cvc4".

By illustrating the dart example ([19], Chapter 2, page 21), which serves as a counter model, Horty disproves the following two formulas:

- $\varphi \rightarrow \Diamond[a\ cstit : \varphi]$
- $\Diamond[a\ cstit : \varphi \vee \psi] \rightarrow (\Diamond[a\ cstit : \varphi] \vee \Diamond[a\ cstit : \psi])$

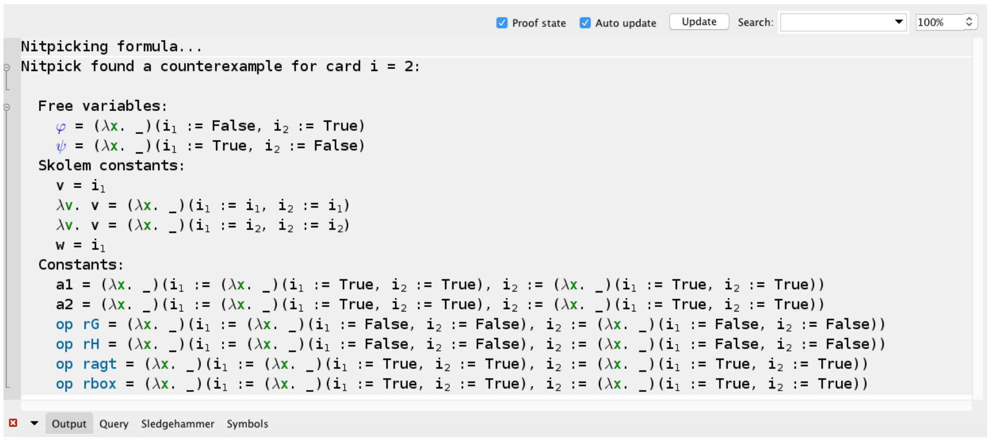


```
(* Axioms/Laws from Horty's Book *)

(* Page 17 *)

lemma "⌊φ ≡ ψ⌋ ⟹ ⌊(a1 cstit φ) ≡ (a1 cstit ψ)⌋ "
  by simp

lemma "⌊ a1 cstit ⊤ ⌋" sledgehammer
  by simp

lemma "⌊ (a1 cstit (φ ∧ ψ))  → ((a1 cstit (φ)) ∧ (a1 cstit (ψ))) ⌋ "
  by simp

lemma "⌊ ((a1 cstit (φ)) ∧ (a1 cstit (ψ))) → (a1 cstit (φ ∧ ψ)) ⌋ "
  by simp

(* Page 18 *)
lemma  "⌊¬ (a1 dstit ⊤) ⌋"
  by simp
```

```
Sledgehammering...
Proof found...
"z3": Try this: by simp (0.6 ms)
"cvc4": Try this: by simp (0.5 ms)
```

Figure 3.1: Proofs generated by Sledgehammer

When formulating those formulas into *Isabelle/HOL*, $Nitpick$ is also able to find a counter model for both formulas. Counter models are printed out in the console of *Isabelle/HOL*. The output for the second formula is illustrated in figure 3.2.



```
Nitpicking formula...
Nitpick found a counterexample for card i = 2:

  Free variables:
    φ = (λx. _)(i₁ := False, i₂ := True)
    ψ = (λx. _)(i₁ := True, i₂ := False)
  Skolem constants:
    v = i₁
    λv. v = (λx. _)(i₁ := i₁, i₂ := i₁)
    λv. v = (λx. _)(i₁ := i₂, i₂ := i₂)
    w = i₁
  Constants:
    a1 = (λx. _)(i₁ := (λx. _)(i₁ := True, i₂ := True), i₂ := (λx. _)(i₁ := True, i₂ := True))
    a2 = (λx. _)(i₁ := (λx. _)(i₁ := True, i₂ := True), i₂ := (λx. _)(i₁ := True, i₂ := True))
    op rG = (λx. _)(i₁ := (λx. _)(i₁ := False, i₂ := False), i₂ := (λx. _)(i₁ := False, i₂ := False))
    op rH = (λx. _)(i₁ := (λx. _)(i₁ := False, i₂ := False), i₂ := (λx. _)(i₁ := False, i₂ := False))
    op ragt = (λx. _)(i₁ := (λx. _)(i₁ := True, i₂ := True), i₂ := (λx. _)(i₁ := True, i₂ := True))
    op rbox = (λx. _)(i₁ := (λx. _)(i₁ := True, i₂ := True), i₂ := (λx. _)(i₁ := True, i₂ := True))
```

Figure 3.2: Counter model generated by Nitpick

$Nitpick$ has found a counter example for card i=2, which means that the generated model consists of two possible worlds i_1 and i_2. Next, we have the free variables φ and ψ where φ is false at i_1 and true at i_2, and ψ holds in i_1 but it does not in i_2. For this kind of counter model the Skolem constants are not important and can therefore be ignored. Then we move to the constants which correspond to the different accessibility relations. $a1$ and $a2$ are representing the relations R_1 for agent 1 and R_2 for agent 2, respectively. For the relation R_1 and R_2, $Nitpick$ relates the two worlds i_1 and i_2 as follows, we have that $R_1(i_1) = R_1(i_2) = \{i_1, i_2\}$ and $R_2(i_1) = R_2(i_2) = \{i_1, i_2\}$. The constants rG and rH represents the relations

R_G and R_H for the temporal operators G and H. In this model, they are not needed and therefore $R_G = R_H = \varnothing$. Finally, the constant $rbox$ represents the relation $R_\Box$ for the necessity operator $\Box$ and the constant $ragt$ represents the relation R_{Agt} for the group stit operator $[Agt]$. Just like for the relations R_1 and R_2, we have that $R_\Box = R_{Agt} = \{i_1, i_2\}$. In terms of Kripke semantics for temporal STIT logic, the generated counter model looks as follows:

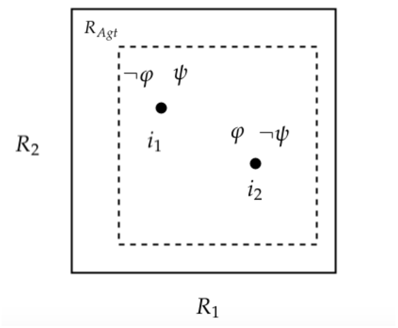


Figure 3.3: Generated counter model in Kripke semantics

In the illustrated figure 3.3, it can easily be verified that the antecedent of the formula holds whereas the conclusion does not, which falsifies the second formula. Even though the purpose of this counter model was to disprove the second formula, it shows as well the invalidity of the first formula.

Other axioms and laws from Horty's book were also encoded into *Isabelle/HOL* and could all be proven by $Sledgehammer$ except for one axiom where a proof could be found but the proof reconstruction failed.

3.4.9 Axioms - Lorini

In temporal STIT logic, the K axiom holds for the $\Box$ operator, for the group STIT operator, for every individual STIT operator, as well as for the temporal operator G and H. We can formulate the K-axioms for each operator as follows:

- $\lfloor((\Box\varphi) \wedge (\Box(\varphi \rightarrow \psi))) \rightarrow (\Box\psi)\rfloor$
- $\lfloor((gstit\,\varphi) \wedge (gstit\,(\varphi \rightarrow \psi))) \rightarrow (gstit\,\psi)\rfloor$
- $\lfloor((a1\,cstit\,\varphi) \wedge (a1\,cstit\,(\varphi \rightarrow \psi))) \rightarrow (a1\,cstit\,\psi)\rfloor$
- $\lfloor((G\varphi) \wedge (G(\varphi \rightarrow \psi))) \rightarrow (G\psi)\rfloor$
- $\lfloor((H\varphi) \wedge (H(\varphi \rightarrow \psi))) \rightarrow (H\psi)\rfloor$

All of the above axioms were easily verified by the $Sledgehammer$ tool. Next, we have the rule of necessitation which holds also for every operator.

- $\lfloor \varphi \rfloor \Longrightarrow \lfloor (\Box \varphi) \rfloor$
- $\lfloor \varphi \rfloor \Longrightarrow \lfloor (gstit\, \varphi) \rfloor$
- $\lfloor \varphi \rfloor \Longrightarrow \lfloor (a1\, cstit\, \varphi) \rfloor$
- $\lfloor \varphi \rfloor \Longrightarrow \lfloor (G \varphi) \rfloor$
- $\lfloor \varphi \rfloor \Longrightarrow \lfloor (H \varphi) \rfloor$

These axioms can also be directly derived and require no further attention. Furthermore, we have axiom 4 for all the operators except H.

- $\lfloor (\Box \varphi) \rightarrow (\Box (\Box \varphi)) \rfloor$
- $\lfloor (gstit\, \varphi) \rightarrow (gstit\, (gstit\, \varphi)) \rfloor$
- $\lfloor (a1\, cstit\, \varphi) \rightarrow (a1\, cstit\, (a1\, cstit\, \varphi)) \rfloor$
- $\lfloor (G \varphi) \rightarrow (G (G \varphi)) \rfloor$

Those axioms follow from the facts that $R_\Box$, R_{Agt} and every R_i are equivalence relations and that R_G is a transitive relation as well. Additionally, we have axiom T for the $\Box$ operator, for the group STIT operator and every individual STIT operator.

- $\lfloor (\Box \varphi) \rightarrow \varphi \rfloor$
- $\lfloor (gstit\, \varphi) \rightarrow \varphi \rfloor$
- $\lfloor (a1\, cstit\, \varphi) \rightarrow \varphi \rfloor$

Since $R_\Box$, R_{Agt} and every R_i are equivalence relations, they are also reflexive which is necessary to prove that axiom T holds for those operators. Next, we have the axiom B for the same operators as for axiom T.

- $\lfloor \varphi \rightarrow (\Box (\neg (\Box (\neg \varphi)))) \rfloor$
- $\lfloor \varphi \rightarrow (gstit\, (\neg (gstit\, (\neg \varphi)))) \rfloor$
- $\lfloor \varphi \rightarrow (a1\, cstit\, (\neg (a1\, cstit\, (\neg \varphi)))) \rfloor$

Similar to axiom T, axiom B follows from the fact that the relations are symmetric due to the equivalence property. Moreover, we have the axiom D for the operator G:

- $\lfloor \neg((G\varphi) \wedge (G\neg\varphi)) \rfloor$

This axiom can be derived due the seriality property of the relation R_G. The axioms $Conv_{G,H}$ and $Conv_{H,G}$, which are the basic interaction axioms between future and past of minimal tense logic, are the following:

- $\lfloor \varphi \rightarrow (G(P\varphi)) \rfloor$
- $\lfloor \varphi \rightarrow (H(F\varphi)) \rfloor$

Both axioms can be proven by the fact that R_H is defined as the inverse relation of R_G. To guarantee the linearity of the future resp. the past, we have the axioms $Connected_G$ resp. $Connected_H$ and those can be formulated as:

- $\lfloor (P(F\varphi) \rightarrow ((P\varphi) \vee \varphi \vee (F\varphi)) \rfloor$
- $\lfloor (F(P\varphi) \rightarrow ((P\varphi) \vee \varphi \vee (F\varphi)) \rfloor$

The constraints $C4$ resp. $C5$ together with the fact that R_H is the inverse relation of R_G are used to derive those axioms. Further we have $(\Box \rightarrow i)$, $(i \rightarrow Agt)$ and (AIA), which denote the central principles in Xu's axiomatization in STIT logic. Since we only consider two agents, those axioms can be defined as:

- $\lfloor (\Box\varphi) \rightarrow (a1\, cstit\, \varphi) \rfloor$
- $\lfloor ((a1\, cstit\, \varphi) \wedge (a2\, cstit\, \psi)) \rightarrow (gstit\, (\varphi \wedge \psi)) \rfloor$
- $\lfloor ((\Diamond(a1\, cstit\, \varphi)) \wedge (\Diamond(a2\, cstit\, \psi))) \rightarrow (\Diamond((a1\, cstit\, \varphi) \wedge (a2\, cstit\, \psi))) \rfloor$

The first axiom can be proven by constraint $C1$ whereas the second one follows from the constraint $C3$. The $Sledgehammer$ tool is not able to find a proof for the third axiom but the countermodel generator $Nitpick$ also could not find a counter model for this axiom. To complete the axiomatization system, we have modus ponens (MP), axiom $(NCUH)$ and (IRR), a variant of Gabbay's irreflexivity rule.

- $\lfloor (F(\Diamond\varphi)) \rightarrow (dualgstit(F\varphi)) \rfloor$
- $(\lfloor \varphi \rfloor) \wedge (\lfloor \varphi \rightarrow \psi \rfloor) \Longrightarrow \lfloor \psi \rfloor$
- $\lfloor ((\Box(\neg p)) \wedge (\Box((G\, p) \wedge (H\, p)))) \rightarrow \varphi \rfloor \Longrightarrow \lfloor \varphi \rfloor$

Modus pones is proven automatically and $(NCUH)$ can be derived from constraint $C6$. Unfortunately, axiom (IRR) could not be proven by the $Sledgehammer$ tool.

3.5 Limitations of Isabelle/HOL

The fact that R_G is a serial relation combined with the constraint $(C7)$, makes every temporal Kripke STIT model infinite. In other words, it consists of infinitely many worlds.

This has been noticed after we were running a step by step consistency check in Isabelle/HOL, meaning that we first let $Nitpick$ find a model satisfying the constraint $C1$. Whenever a model was found, we added the next constraint and looked for another model until all of the seven constraints were covered. However, $Nitpick$ has not been able to find a model satisfying all of them. Figure 3.4 illustrates the model, found by $Nitpick$, satisfying the constraints from $(C1)$ until $(C6)$, but not constraint $(C7)$.

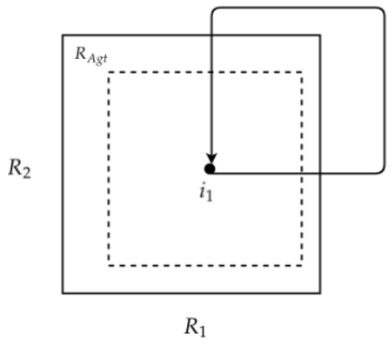


Figure 3.4: Model satisfying constraints C1-C6
but falsifying constraint C7

But just because $Nitpick$ is not able to find a model satisfying all seven constraints, it does not mean that such a model does not exist. $Nitpick$ only works for finite models, thus it just means that there is no finite model satisfying all the constraints and that the constraint $(C7)$ together with R_G, defined as a serial relation, can never hold in a finite model. However, there still exist infinite models satisfying all of the constraints and in order to realize the implementation of the embedding in Isabelle/HOL, we decided to drop the seriality property of R_G.

By removing the seriality condition from R_G, $Nitpick$ is able to generate a finite model satisfying all of the constraints. As a consequence, some of the axioms from Lorini's work [22] are no longer valid since they can only be applied to infinite models. For instance, axiom D is invalid for the G operator since it followed from the seriality property of R_G. The axiom (IRR) could also not be proven by $Sledgehammer$ and instead, a counter model was found. Further, Lorini proved the validity of the following two formulas:

- $G\Diamond G^*\varphi \rightarrow \langle Agt\rangle G\varphi$

– $G\Diamond(G^*\varphi \wedge F^*\psi) \rightarrow \langle Agt\rangle(G\varphi \wedge F\psi)$

However, when translated into Isabelle/HOL in order to verify them, *Sledgehammer* was not able to provide a proof for both and for the later formula, a counter model was found.

3.6 T-STIT I/O logic in HOL

T-STIT I/O logic uses formulas from T-STIT logic. We showed earlier in this chapter how to formulate those in HOL. The next task is to define the operators out_1 and out_2 in HOL and implemented them in *Isabelle/HOL* so that we are able to verify some examples of T-STIT I/O logic within the framework. In this section, we describe the embeddings of the I/O operators out_1 and out_2. The literature already provides us with a semantical embedding of the out_1 operator in HOL [7]. We start off by showing how Parent and Benzmüller [7] embedded the out_1 operator in HOL and how they implemented it in *Isabelle/HOL*. They considered propositional logic as the base logic for the out_1 operator. But we show that their implementation can be used together with the previously described implementation of T-STIT logic, so that we are able to check examples of T-STIT I/O logic in Isabelle/HOL. Next, we present the embedding of the out_2 operator in HOL. At first, we also consider propositional logic as the base logic for the operator and change it then to T-STIT logic.

3.6.1 Simple minded output out_1

Parent and Benzmüller [7] present their first attempt of a semantical embedding of I/O logic in HOL with the main focus lying on the I/O operator out_1. In particular, it documents their initial attempt to embed I/O in *Isabelle/HOL*, which failed due to an inappropriate encoding of the statement $\vDash \varphi$, as well as a proper embedding for the operator out_1 in HOL.

For a given set of conditional norms N and a set of facts A, the I/O operator out_1 is defined as follows $out_1(N, A) = Cn(N(Cn(A)))$ with $Cn(A) = \{x \mid A \vDash x\}$ and $N(A) = \{x \mid (a, x) \in N$ for some $a \in A\}$. To achieve a proper formulation of the operator out_1, Parent and Benzmüller used the technique of lifting propositional formulas of I/O logic to predicates on possible worlds. This means that they are mapped to HOL terms of type $i \rightarrow o$ where i denotes the type of possible worlds and o refers to the boolean type. The propositional logical connectives, such as $\neg$, $\wedge$, $\wedge$ and $\supset$, are also lifted and are defined in exactly the same way as in section 5.4.2. However, the reason to apply this technique in this case, is to use $valid\ \varphi$, which is denoted by $\lfloor\varphi\rfloor$, as a suitable encoding of $\vDash \varphi$.

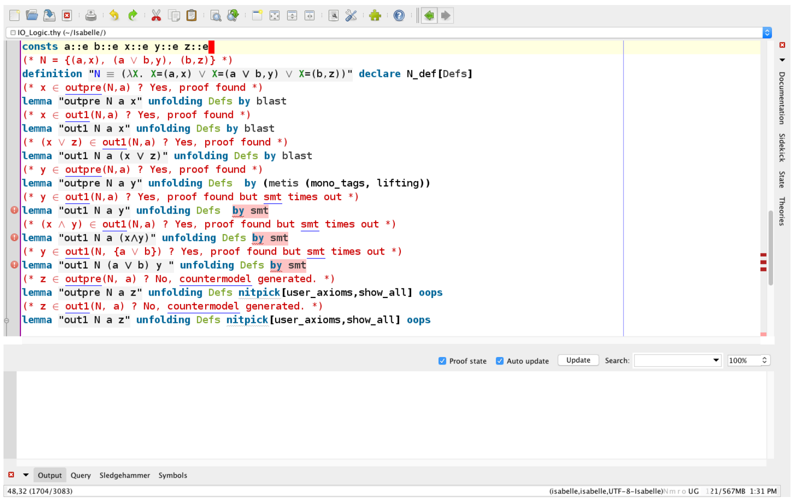
After having found an appropriate encoding for $\vDash \varphi$, the operator $outpre$ was defined such that $outpre(N,A) = N(Cn(A))$. Thus, $outpre(N,A)$ denotes the set $\{y \in \mathcal{L} \mid \exists f \in Cn(A)$ such that $(f,y) \in N\}$ where $\mathcal{L}$ is the set containing all the formulas of propositional logic. To obtain the operator out_1, $outpre$ has to be closed under logical consequence. Closing a set under logical consequence results in an infinite set, therefore only an approximation of out_1 was defined. In particular, $out_1(N,A)$ is restricted to the consequences that follow from maximally three formulas in $outpre(N,A)$. Thus, $out_1(N,A)$ denotes the set $\{x \in \mathcal{L} \mid \{i,j,k\} \subseteq outpre(N,A)$ and $\vDash (i \wedge j \wedge k) \supset x\}$. Another thing to keep in mind is that the set of facts A is limited to singleton sets only. So out_1 is adapted for the use of a singleton formula such as a, but one also may consider a as the conjunction of all the formulas in a set A. More precisely, $a = a_1 \wedge \cdots \wedge a_n$ with $a_i \in A$ for $i \in \{1 \dots n\}$. This has been done due to technical reasons. In *Isabelle/HOL*, the operators $outpre$ and out_1 are then formulated as follows:

- $outpre = \lambda N.\lambda a.\lambda y.\exists f.(\lfloor a \supset f \rfloor \wedge N(f,y))$
- $out1 = \lambda N.\lambda a.\lambda x.(\exists i\, j\, k.\, outpre\, N\, a\, i \wedge outpre\, N\, a\, j \wedge outpre\, N\, a\, k \wedge \lfloor (i \wedge j \wedge k) \supset x \rfloor)$

Parent and Benzmüller defined the operations $outpre$ and $out1$ for formulas of propositional logic. In our case, we are using T-STIT logic as the base logic for our I/O operators and we used the same technique to represent those formulas in HOL. The encoding of validity for formulas of propositional logic is exactly the same as validity for formulas of T-STIT logic. This means that those defined operators can also be applied when we are changing $\mathcal{L}$ to $\mathcal{L}_{\text{T-STIT}}$.

Having defined the operators $outpre$ and $out1$, we are able to analyze some examples. For instance, example 2 from chapter 2 can be formulated in *Isabelle/HOL*. We have that $N = \{(a,x),(a \vee b,y),(b,z)\}$ and $A = \{a\}$. See figure 3.5.

First we have to declare the propositions a, b, x, y and z. The propositions are mapped to associated HOL terms of type $i \to o$. This type represents a function type from possible worlds to truth values in HOL. This type is abbreviated here as e. Next, we define the set of conditional norms N and we check first if x is in $outpre(N,a)$, meaning in $N(Cn(a))$, and afterwards if x is in $out_1(N,a)$. In both cases, $Sledgehammer$ is able to find a proof. Further, $Sledgehammer$ provided proofs to show that $(x \vee z) \in out_1(N,a)$ and $y \in outpre(N,a)$. For the statements that y and $(x \wedge y)$ are also in $out_1(N,a)$, we also get a proof, however, the proof solver times out. The same problem arises when checking if y is in the $out_1(N,\{a \vee b\})$. Finally, countermodels could be found to show that z is neither in $outpre(N,a)$ nor in $out_1(N,a)$. Countermodels are generated



IO_Logic.thy (~/Isabelle/)

```
consts a::e b::e x::e y::e z::e
(* N = {(a,x), (a ∨ b,y), (b,z)} *)
definition "N ≡ (λX. X=(a,x) ∨ X=(a ∨ b,y) ∨ X=(b,z))" declare N_def[Defs]
(* x ∈ outpre(N,a) ? Yes, proof found *)
lemma "outpre N a x" unfolding Defs by blast
(* x ∈ out1(N,a) ? Yes, proof found *)
lemma "out1 N a x" unfolding Defs by blast
(* (x ∨ z) ∈ out1(N,a) ? Yes, proof found *)
lemma "out1 N a (x ∨ z)" unfolding Defs by blast
(* y ∈ outpre(N,a) ? Yes, proof found *)
lemma "outpre N a y" unfolding Defs  by (metis (mono_tags, lifting))
(* y ∈ out1(N,a) ? Yes, proof found but smt times out *)
lemma "out1 N a y" unfolding Defs  by smt
(* (x ∧ y) ∈ out1(N,a) ? Yes, proof found but smt times out *)
lemma "out1 N a (x∧y)" unfolding Defs by smt
(* y ∈ out1(N, {a ∨ b}) ? Yes, proof found but smt times out *)
lemma "out1 N (a ∨ b) y " unfolding Defs by smt
(* z ∈ outpre(N, a) ? No, countermodel generated. *)
lemma "outpre N a z" unfolding Defs nitpick[user_axioms,show_all] oops
(* z ∈ out1(N, a) ? No, countermodel generated. *)
lemma "out1 N a z" unfolding Defs nitpick[user_axioms,show_all] oops
```

Figure 3.5: Isabelle/HOL: out_1 example

for the HOL formula of the respective statement. For instance, the statement $z \in outpre(N, a)$ unfolds as follows. We start from the line:

$$outpre\, N\, a\, z$$

Then, $outpre$ will be substituted with its formulation and we get:

$$(\lambda N.\lambda a.\lambda y.\exists f.(\lfloor a \supset f \rfloor \wedge N(f, y)))\, N\, a\, z$$

During the next step, the λ-calculus will be performed. Thus, N, a and y will be substituted for N, a and z, respectively, and we obtain the following expression:

$$\exists f.(\lfloor a \supset f \rfloor \wedge N(f, z))$$

Next, at first N will be replaced by its definition. Then we end up again with an expression containing a λ-calculus operation. All X's in the expression will be substituted for (f, z). Thus we have the following lines:

$$\exists f.(\lfloor a \supset f \rfloor \wedge (\lambda X.\, X = (a, x) \vee X = (a \vee b, y) \vee X = (b, z))(f, z))$$

$$\exists f.(\lfloor a \supset f \rfloor \wedge ((f, z) = (a, x) \vee (f, z) = (a \vee b, y) \vee (f, z) = (b, z)))$$

As the propositions are of type e, the connectives such as $\vee$ are lifted as well. N contains the element $(a \vee b, y)$, and thus we have to replace the lifted connective by its expression.

$$\exists f.(\lfloor a \supset f \rfloor \wedge ((f, z) = (a, x) \vee (f, z) = (\lambda w.a(w) \vee b(w), y) \vee (f, z) = (b, z)))$$

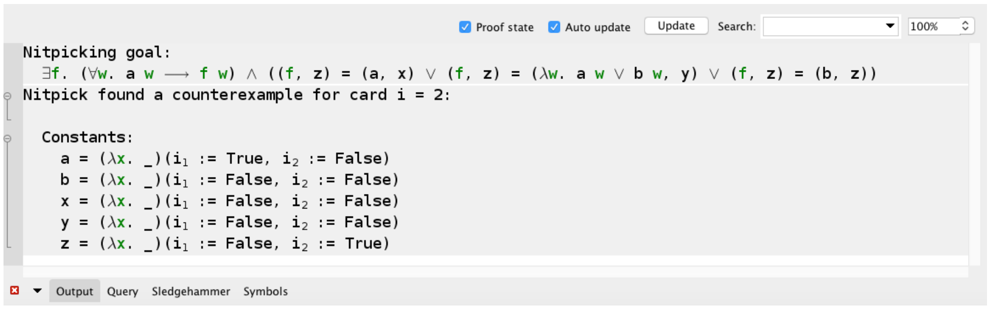
Finally the last step deals with the expression $\lfloor a \supset f \rfloor$. First, the notation of $\lfloor \cdot \rfloor$ will substituted with its definition and the same thing will be done for lifted implication $\supset$.

$$\exists f.((\forall w.(a \supset f)(w)) \wedge ((f,z) = (a,x) \vee (f,z) = (\lambda w.a(w) \vee b(w), y) \vee (f,z) = (b,z)))$$

$$\exists f.((\forall w.(\lambda w.a(w) \longrightarrow f(w))(w)) \wedge ((f,z) = (a,x) \vee (f,z) = (\lambda w.a(w) \vee b(w), y) \vee (f,z) = (b,z)))$$

After a final λ-calculus operation, the unfolding is complete. Thus $z \in outpre(N,a)$ unfolds into the following formula:

$$\exists f.((\forall w.a(w) \longrightarrow f(w)) \wedge ((f,z) = (a,x) \vee (f,z) = (\lambda w.a(w) \vee b(w), y) \vee (f,z) = (b,z)))$$



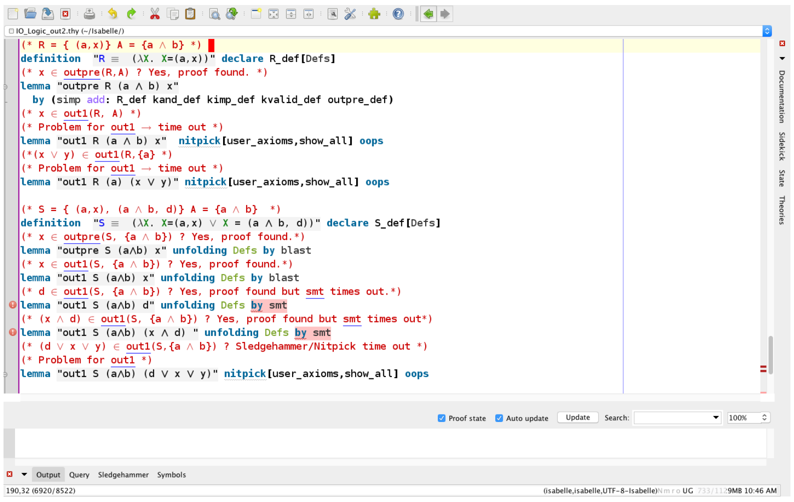
```
Nitpicking goal:
  ∃f. (∀w. a w ⟶ f w) ∧ ((f, z) = (a, x) ∨ (f, z) = (λw. a w ∨ b w, y) ∨ (f, z) = (b, z))
Nitpick found a counterexample for card i = 2:

  Constants:
    a = (λx. _)(i₁ := True, i₂ := False)
    b = (λx. _)(i₁ := False, i₂ := False)
    x = (λx. _)(i₁ := False, i₂ := False)
    y = (λx. _)(i₁ := False, i₂ := False)
    z = (λx. _)(i₁ := False, i₂ := True)
```

Figure 3.6: Isabelle/HOL: Countermodel for $z \in outpre(N,a)$

For this formula, $Nitpick$ is able to give a countermodel and therefore disproves the statements. Figure 3.6 shows a console output of a countermodel in *Isabelle/HOL*. Consider two possible worlds i_1 and i_2. We have that the proposition a is true at i_1 but false at i_2 and so a corresponds to the set $\{i_1\}$. Further we have that the propositions b, x and y are equal to the empty set since they are all false at both worlds and z is the set $\{i_2\}$ since z holds at the world i_2. The expression $\lambda w.a(w) \vee b(w)$ also denotes the set $\{i_1\}$. It is then impossible to find f such that this formula evaluates to true.

When using a as the conjunction of all the elements in a set of facts A, the scope of the search space for the tools $Sledgehammer$ and $Nitpick$ seems to be a bit problematic for this embedding of out_1. Figure 3.7 is illustrating this problems.

In the first example, we have a set of conditional norms $R = \{a, x\}$ and a set $A = \{a \wedge b\}$. In *Isabelle/HOL*, it could be proven that x is in $outpre(R,A)$, however, $Sledgehammer$ times out when trying to prove that x is in $out_1(R,A)$. When changing the set A to the singleton set $\{a\}$, $Sledgehammer$ is also not able to give a proof for showing that $(x \vee y)$ is in $out_1(R,a)$.



```
(* R = { (a,x)} A = {a ∧ b} *)
definition  "R ≡  (λX. X=(a,x))" declare R_def[Defs]
(* x ∈ outpre(R,A) ? Yes, proof found. *)
lemma "outpre R (a ∧ b) x"
  by (simp add: R_def kand_def kimp_def kvalid_def outpre_def)
(* x ∈ out1(R, A) *)
(* Problem for out1 → time out *)
lemma "out1 R (a ∧ b) x"  nitpick[user_axioms,show_all] oops
(*(x ∨ y) ∈ out1(R,{a} *)
(* Problem for out1 → time out *)
lemma "out1 R (a) (x ∨ y)" nitpick[user_axioms,show_all] oops

(* S = { (a,x), (a ∧ b, d)} A = {a ∧ b}  *)
definition  "S ≡  (λX. X=(a,x) ∨ X = (a ∧ b, d))" declare S_def[Defs]
(* x ∈ outpre(S, {a ∧ b}) ? Yes, proof found.*)
lemma "outpre S (a∧b) x" unfolding Defs by blast
(* x ∈ out1(S, {a ∧ b}) ? Yes, proof found.*)
lemma "out1 S (a∧b) x" unfolding Defs by blast
(* d ∈ out1(S, {a ∧ b}) ? Yes, proof found but smt times out.*)
lemma "out1 S (a∧b) d" unfolding Defs by smt
(* (x ∧ d) ∈ out1(S, {a ∧ b}) ? Yes, proof found but smt times out*)
lemma "out1 S (a∧b) (x ∧ d) " unfolding Defs by smt
(* (d ∨ x ∨ y) ∈ out1(S,{a ∧ b}) ? Sledgehammer/Nitpick time out *)
(* Problem for out1 *)
lemma "out1 S (a∧b) (d ∨ x ∨ y)" nitpick[user_axioms,show_all] oops
```

Figure 3.7: Isabelle/HOL: Problems for out_1

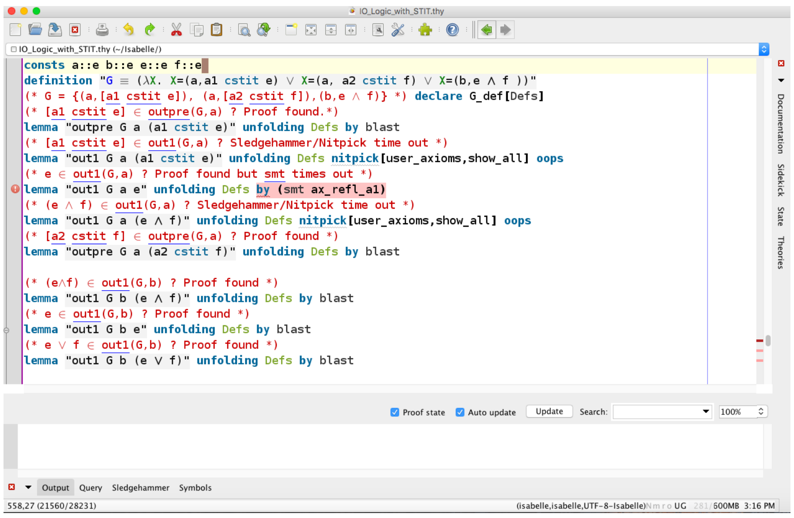
For the second example, it could be shown that x is in $outpre(S, \{a \wedge b\})$ as well as in $out_1(S, \{a \wedge b\})$ where $S = \{(a, x), (a \wedge b, d)\}$. Also, proofs for showing that d and $(x \wedge d)$ are in $out_1(S, \{a \wedge b\})$ could be found, but the proof solver times out. Unfortunately, the search space is too large for $Sledgehammer$ and therefore is not able to show that $(d \vee x \vee y)$ is in $out_1(S, \{a \wedge b\})$.

As illustrated on different examples, the $Sledgehammer$ tool only managed to find proofs when dealing with propositional formulas of low complexity. By increasing a formula's complexity, the search space becomes too large which then results in a timeout of $Sledgehammer$. So far, we did not use any formulas containing any STIT operators. When doing so, the problem of the large search scope for the tools such as $Sledgehammer$ and $Nitpick$ still remains. See Figure 3.8).

Consider the set of conditional norms $G = \{(a, [\alpha]e), (a, [\beta]f), (b, e \wedge f)\}$ and the set of facts $A = \{a\}$ where a, b, e, f are propositional letters and α, β are two agents.

A	$Cn(A)$	$G(Cn(A))$	$out_1(G, A)$
a	$Cn(a)$	$\{[\alpha]e, [\beta]f\}$	$Cn(\{[\alpha]e, [\beta]f\})$
b	$Cn(b)$	$\{e \wedge f\}$	$Cn(\{e \wedge f\})$

We have that $out_1(G, a) = Cn(\{[\alpha]e, [\beta]f\})$, thus formulas such as $[\alpha]e$, $[\beta]f$, e, f and $(e \wedge f)$ are all included in $out_1(G, a)$. When formulating this example in *Isabelle/HOL*, only two proofs could be provided by $Sledgehammer$. In particular, it showed that $[\alpha]e$ and $[\beta]f$ are in $outpre(G, a)$. All the other statements were beyond the search scope of $Sledgehammer$ and therefore no proofs were found. When choosing the singleton b as the set of facts, all statements



```
consts a::e b::e e::e f::e
definition "G ≡ (λX. X=(a,a1 cstit e) ∨ X=(a, a2 cstit f) ∨ X=(b,e ∧ f ))"
(* G = {(a,[a1 cstit e]), (a,[a2 cstit f]),(b,e ∧ f)} *) declare G_def[Defs]
(* [a1 cstit e] ∈ outpre(G,a) ? Proof found.*)
lemma "outpre G a (a1 cstit e)" unfolding Defs by blast
(* [a1 cstit e] ∈ out1(G,a) ? Sledgehammer/Nitpick time out *)
lemma "out1 G a (a1 cstit e)" unfolding Defs nitpick[user_axioms,show_all] oops
(* e ∈ out1(G,a) ? Proof found but smt times out *)
lemma "out1 G a e" unfolding Defs by (smt ax_refl_a1)
(* (e ∧ f) ∈ out1(G,a) ? Sledgehammer/Nitpick time out *)
lemma "out1 G a (e ∧ f)" unfolding Defs nitpick[user_axioms,show_all] oops
(* [a2 cstit f] ∈ outpre(G,a) ? Proof found *)
lemma "outpre G a (a2 cstit f)" unfolding Defs by blast

(* (e∧f) ∈ out1(G,b) ? Proof found *)
lemma "out1 G b (e ∧ f)" unfolding Defs by blast
(* e ∈ out1(G,b) ? Proof found *)
lemma "out1 G b e" unfolding Defs by blast
(* e ∨ f ∈ out1(G,b) ? Proof found *)
lemma "out1 G b (e ∨ f)" unfolding Defs by blast
```

Figure 3.8: Isabelle/HOL: out_1 and STIT logic

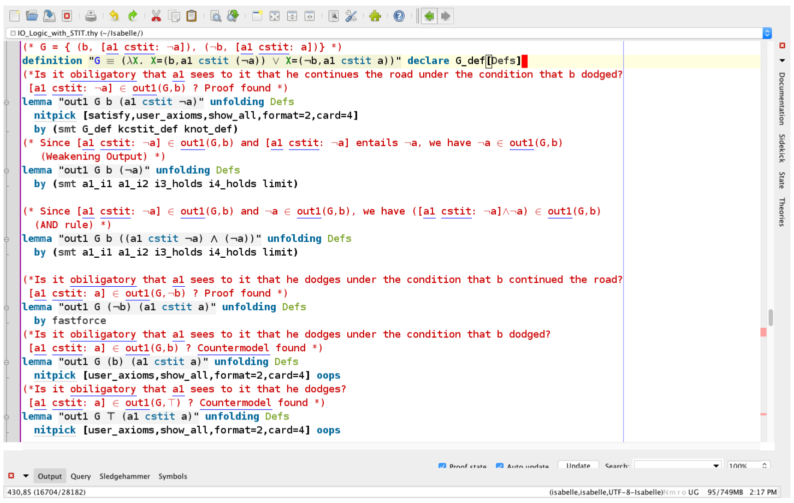
were proved, however, they were of small interest since they did not include a STIT operator.

A possibility to reduce the search space of *Sledgehammer* is to directly formulate a particular model in *Isabelle/HOL*. For instance, we used Horty's model for the driving example ([19], Chapter 5, pages 119-121). The original model is given in $BT + AC$ structured, but we translated it into a temporal Kripke STIT model and specified this model in *Isabelle/HOL*. The example presents a situation where we have two drivers who drive toward each other on a one-lane road. They have no possibility to stop or to communicate with each other, and at one particular moment, each one of them must independently decide whether he continues driving along the road or swerves. Further, the drivers can only swerve in one single direction, therefore they will end up in a collision when both of them choose to continue driving along the road or to swerve. A collision can only be avoided when one of them swerves and the other does not. We formulated then a set of conditional norms G containing the following two pairs $(b, [\alpha]\neg a)$ and $(\neg b, [\alpha]a)$. Formally, we have then $G = \{(b, [\alpha]\neg a), (\neg b, [\alpha]a)\}$. The former expresses that under the condition that agent β swerved, it is obligatory that agent α sees to it that he continues driving the road whereas the latest one states that under the condition that agent β continued driving the road, it is obligatory that agent α sees to it that he swerves. Applying out_1 to the set G and using b as the input situation, it should then be obligatory that $[\alpha]\neg a$. In fact, we have:

A	$Cn(A)$	$G(Cn(A))$	$out_1(G,A)$
b	$Cn(b)$	$\{[\alpha]\neg a\}$	$Cn(\{[\alpha]\neg a\})$

Obviously, $[\alpha]\neg a \in out_1(G,b)$. Further since $[\alpha]\neg a$ implies $\neg a$, we also have that $(\neg a) \in out_1(G,b)$. Indeed, this is a consequence of the *Weakening the output (WO)* rule. By the *conjunction of the output (AND)* rule, we also have that $([\alpha]\neg a \wedge (\neg a)) \in out_1(G,b)$. When providing the model in *Isabelle/HOL*, $Sledgehammer$ finds in fact proofs to verify that those three formulas are in $out_1(G,b)$. This is depicted in Figure 3.9.

Remark Please note that in the *Isabelle/HOL* implementation, the agents $a1$ and $a2$ represent the drivers α and β, respectively.



```
(* G = { (b, [a1 cstit: ¬a]), (¬b, [a1 cstit: a])} *)
definition "G ≡ (λX. X=(b,a1 cstit (¬a)) ∨ X=(¬b,a1 cstit a))" declare G_def[Defs]
(*Is it obiligatory that a1 sees to it that he continues the road under the condition that b dodged?
 [a1 cstit: ¬a] ∈ out1(G,b) ? Proof found *)
lemma "out1 G b (a1 cstit ¬a)" unfolding Defs
  nitpick [satisfy,user_axioms,show_all,format=2,card=4]
  by (smt G_def kcstit_def knot_def)
(* Since [a1 cstit: ¬a] ∈ out1(G,b) and [a1 cstit: ¬a] entails ¬a, we have ¬a ∈ out1(G,b)
   (Weakening Output) *)
lemma "out1 G b (¬a)" unfolding Defs
  by (smt a1_i1 a1_i2 i3_holds i4_holds limit)

(* Since [a1 cstit: ¬a] ∈ out1(G,b) and ¬a ∈ out1(G,b), we have ([a1 cstit: ¬a]∧¬a) ∈ out1(G,b)
  (AND rule) *)
lemma "out1 G b ((a1 cstit ¬a) ∧ (¬a))" unfolding Defs
  by (smt a1_i1 a1_i2 i3_holds i4_holds limit)

(*Is it obiligatory that a1 sees to it that he dodges under the condition that b continued the road?
 [a1 cstit: a] ∈ out1(G,¬b) ? Proof found *)
lemma "out1 G (¬b) (a1 cstit a)" unfolding Defs
  by fastforce
(*Is it obiligatory that a1 sees to it that he dodges under the condition that b dodged?
 [a1 cstit: a] ∈ out1(G,b) ? Countermodel found *)
lemma "out1 G (b) (a1 cstit a)" unfolding Defs
  nitpick [user_axioms,show_all,format=2,card=4] oops
(*Is it obiligatory that a1 sees to it that he dodges?
 [a1 cstit: a] ∈ out1(G,⊤) ? Countermodel found *)
lemma "out1 G ⊤ (a1 cstit a)" unfolding Defs
  nitpick [user_axioms,show_all,format=2,card=4] oops
```

Figure 3.9: Driving Example in Isabelle/HOL

Further, we checked that $[\alpha]a \in out_1(G,\neg b)$ which means then that under the condition that agent β continued driving on the road, it is obligatory that agent α sees to that he swerves. When asked whether $[\alpha]a \in out_1(G,b)$ or $[\alpha]a \in out_1(G,\top)$, $Nitpick$ generated counter models, which were unsurprisingly in both case identically to the encoded model.

3.6.2 Basic output out_2

In contrast to out_1, the operator out_2 supports reasoning by cases. For a given set of conditional norms N and a set of facts A, we formally write $out_2(N,A) = \bigcap\{Cn(N(V)) \mid A \subseteq V, V \text{ complete}\}$. However, for the embedding, we considered the modal logical formulation of out_2. This allows us to reuse the same techniques which have already been applied to embed different kinds of modal logics in HOL. We will focus first on an embedding for the out_2 with propositional logic as the base logic. According to the modal logical formulation, we have that $x \in out_2(N,A)$ if and only if

i. $x \in Cn(N(\mathcal{L}))$ and

ii. $N^{\Box} \cup A \vdash_S \Box x$, with $N^{\Box} = \{b \to \Box y \mid (b,y) \in N\}$

To formulate $x \in Cn(N(\mathcal{L}))$ in *Isabelle/HOL*, we start first by defining $N(\mathcal{L})$. In particular, we define $N(\mathcal{L})$ as the set $\{y \in \mathcal{L} \mid \exists f \in \mathcal{L}$ such that $(f,y) \in N\}$. This set gets then closed under the logical consequence and thus we can only define an approximation of $Cn(N(\mathcal{L}))$. Just like for the out_1, we consider only those consequences that follow from maximally three formulas in $N(\mathcal{L})$. So, $Cn(N(\mathcal{L}))$ denotes the set $\{x \in \mathcal{L} \mid \{i,j,k\} \subseteq N(\mathcal{L})$ and $\vDash (i \wedge j \wedge k) \supset x\}$. In *Isabelle/HOL*, we formulate them as:

- $G\_L \equiv \lambda G.\lambda y.\exists f.G(f,y)$
- $out2 \equiv \lambda G.\lambda x.(\exists i\, j\, k.\, G\_L\, G\, i \wedge G\_L\, G\, j \wedge G\_L\, G\, k \wedge \lfloor (i{\wedge}j{\wedge}k){\supset}x \rfloor)$

Remark As already stated in the chapter for I/O logic (2), a set of conditional norms can be denoted by any letter. The letter G was mostly used during the implementation in *Isabelle/HOL*, therefore the formulation uses this letter instead of N.

The next step is to find an encoding for $N^{\Box} \cup A \vdash_S \Box x$ in *Isabelle/HOL*. First, we have that **S** refers to any modal logic such that **K** $\subseteq$ **S** $\subseteq$ **K45**. For the implementation we choose **S** equal to **K45**, so the accessibility relation for the $\Box$ operator is transitive and euclidean. Thus it satisfies the following axioms and rules:

(K) $\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$

(4) $\Box\varphi \to \Box\Box\varphi$

(5) $\Diamond\varphi \to \Box\Diamond\varphi$

(Nec) if $\vdash \varphi$ then $\vdash \Box\varphi$

(MP) if $\vdash \varphi$ and $\vdash \varphi \to \psi$ then $\vdash \psi$

In section 5.4.2, we already showed how to formulate the $\Box$ and $\Diamond$ operators in *Isabelle/HOL*. Further, we demonstrated how to formulate transitivity and declare a transitive relation. Declaring a relation as euclidean can be realized in the same way and euclideaness is defined as follows in *Isabelle/HOL*:

- $euclidean\, r \equiv \forall x\, y\, z.\ ((r\, x\, y) \wedge (r\, x\, z)) \to (r\, y\, z)$

Since **S** is defined now as a **K45** modal logic, $N^{\Box} \cup A \vdash_{K45} \Box x$ means that for a finite subset Y, such that $Y \subseteq N^{\Box} \cup A$, it holds that $(\bigwedge y_i \rightarrow \Box x) \in$ **K45** with y_i denoting the elements of Y. Alternatively, $(\bigwedge y_i \rightarrow \Box x) \in$ **K45** means that $\bigwedge y_i \rightarrow \Box x$ is a valid formula in any **K45** modal logic meaning that one needs to show that $\vDash \bigwedge y_i \rightarrow \Box x$. An appropriate notation for validity in *Isabelle/HOL* has already been introduced in Section 5.4.7 and can therefore be reused in this case.

In I/O logic, obligations are referenced to a given set of conditional norms for some input. Thus one always has to define those sets. For instance, consider the classical example which demonstrates the difference between out_1 and out_2. We have $G = \{(a, x), (b, x)\}$, $A = \{a \vee b\}$ and $x \in out_2(G, A)$ but $x \notin out_1(G, A)$. To verify that $x \in out_2(G, A)$ by using the translation into modal logic, we first have to check that $x \in Cn(G(\mathcal{L}))$. This is clearly the case since $Cn(G(\mathcal{L})) = Cn(x)$. In *Isabelle/HOL*, we would start by defining the set G and then try to get proofs for the statements *G_L G x* and *out2 G x*, to show that $x \in G(\mathcal{L})$ and $x \in Cn(G(\mathcal{L}))$, respectively. As shown in figure 3.10, $Sledgehammer$ provided proofs for those statements.

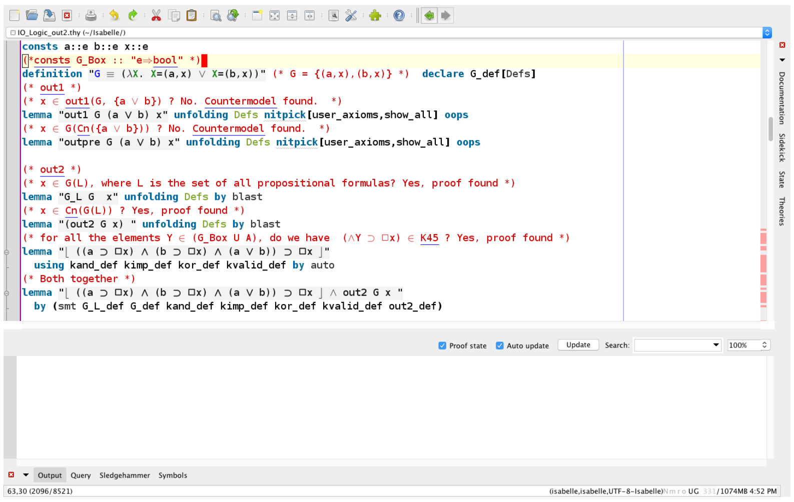

```
consts a::e b::e x::e
(*consts G_Box :: "e⇒bool" *)
definition "G ≡ (λX. X=(a,x) ∨ X=(b,x))" (* G = {(a,x),(b,x)} *)  declare G_def[Defs]
(* out1 *)
(* x ∈ out1(G, {a ∨ b}) ? No. Countermodel found.  *)
lemma "out1 G (a ∨ b) x" unfolding Defs nitpick[user_axioms,show_all] oops
(* x ∈ G(Cn({a ∨ b})) ? No. Countermodel found.  *)
lemma "outpre G (a ∨ b) x" unfolding Defs nitpick[user_axioms,show_all] oops

(* out2 *)
(* x ∈ G(L), where L is the set of all propositional formulas? Yes, proof found *)
lemma "G_L G  x" unfolding Defs by blast
(* x ∈ Cn(G(L)) ? Yes, proof found *)
lemma "(out2 G x) " unfolding Defs by blast
(* for all the elements Y ∈ (G_Box U A), do we have  (∧Y ⊃ □x) ∈ K45 ? Yes, proof found *)
lemma "⌊ ((a ⊃ □x) ∧ (b ⊃ □x) ∧ (a ∨ b)) ⊃ □x ⌋"
  using kand_def kimp_def kor_def kvalid_def by auto
(* Both together *)
lemma "⌊ ((a ⊃ □x) ∧ (b ⊃ □x) ∧ (a ∨ b)) ⊃ □x ⌋ ∧ out2 G x "
  by (smt G_L_def G_def kand_def kimp_def kor_def kvalid_def out2_def)
```

Figure 3.10: Isabelle/HOL: out_1 vs. out_2

For the modal logic part, we focus first on $N^{\Box} \cup A$ which corresponds to the set $\{a \rightarrow \Box x, b \rightarrow \Box x, a \vee b\}$. By compactness, we can always choose the finite subset Y as $N^{\Box} \cup A$. As we want to check that $x \in out_2(G, A)$, we need to show that $((a \rightarrow \Box x) \wedge (b \rightarrow \Box x) \wedge (a \vee b)) \rightarrow \Box x$ is a valid formula in a **K45** modal logic. Assuming that the antecedent is false, then the formula is automatically valid. Assuming that the antecedent holds, then $(a \vee b)$ is true and therefore either a or b is true as well. If a holds, then $\Box x$ must hold as well since we assumed that the antecedent holds. In the same way, $\Box x$ holds in the case b is true. Therefore the formula is valid again. In *Isabelle/HOL*, we check the validity of this formula as follows:

$$\lfloor((a \supset \Box x) \wedge (b \supset \Box x) \wedge (a \vee b)) \supset \Box x\rfloor$$

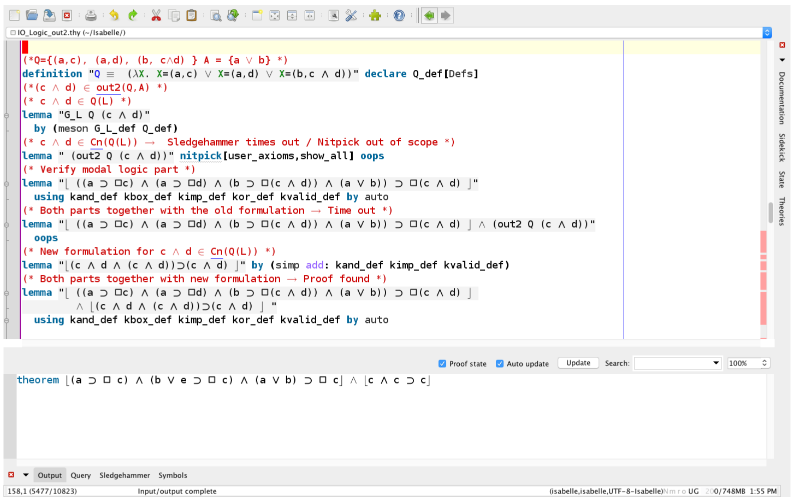
For this statement, *Sledgehammer* also generated a proof, and since we also got a proof for $x \in Cn(G(\mathcal{L}))$, we have that $x \in out_2(G, A)$. We check both statements individually, however one can also check them together by using the HOL conjunction $\wedge$. See figure 3.10. Finally, *Nitpick* generated counter models to show that x is neither in $outpre(G, a)$ nor in $out_1(G, a)$.

Just like for out_1, we also encountered the same problem for out_2 regarding the large search scope for *Sledgehammer*. $x \in out_2(N, A)$ means that $x \in Cn(N(\mathcal{L}))$ and $N^{\Box} \cup A \vdash_{K45} \Box x$. The modal logic part has never caused problems in *Isabelle/HOL* however we could not always get proofs for the first part. Consider a set of conditional norms $Q = \{(a, c), (a, d), (b, c \wedge d)\}$ and $A = \{a \vee b\}$, *Sledgehammer* is able to give a proof for the modal logic part as well as for $(c \wedge d) \in Q(\mathcal{L})$. But when closing $Q(\mathcal{L})$ under the set of logical consequences, it could not prove that $(c \wedge d) \in Cn(Q(\mathcal{L}))$. See figure 3.11 . In *Isabelle/HOL*, we only defined an approximation of $Cn(Q(\mathcal{L}))$. We are looking only at the consequences which follow maximally from three formulas in $Q(\mathcal{L})$. This requires to make use of the quantifier $\exists$ which will look for propositional formulas i, j and k such that they corresponds to some head y of some pair $(b, y) \in Q$ and that $\vDash (i \wedge j \wedge k) \supset (c \wedge d)$. This use of the quantifier $\exists$ may be the reason which caused the time out of *Sledgehammer*. Thus we were trying to find an alternative formulation for $(c \wedge d) \in Cn(Q(L))$ without making use of a quantifier. In *Isabelle/HOL*, we always explicitly define the sets of conditional norms therefore it is always finite and we know which elements it contains. In that case, we could take the conjunction of all the heads y_i for the pairs (b_i, y_i) and directly check if $\vDash \bigwedge y_i \supset (c \wedge d)$ which corresponds to $\lfloor(y_1 \wedge \cdots \wedge y_n) \supset (c \wedge d)\rfloor$ in HOL. For the set Q, we formulate $(c \wedge d) \in Cn(Q(\mathcal{L}))$ as $\lfloor(c \wedge d \wedge (c \wedge d)) \supset (c \wedge d)\rfloor$ and this obviously holds.

Using the formulation mentioned above, *Sledgehammer* is also able to immediately come up with a proof. Thus we have that $(c \wedge d) \in Cn(Q(\mathcal{L}))$. To verify $Q^{\Box} \cup A \vdash_{K45} \Box(c \wedge d)$ in *Isabelle/HOL*, we need to check that

$$\lfloor((a \supset \Box c) \wedge (a \supset \Box d) \wedge (b \supset \Box(c \wedge d)) \wedge (a \vee b)) \supset \Box(c \wedge d)\rfloor$$

As depicted in Figure 3.11, *Sledgehammer* shows that this is a valid formula. Further it is able to give a proof for the conjunction of the statements $(c \wedge d) \in Cn(Q(\mathcal{L}))$ and $Q^{\Box} \cup A \vdash_{K45} \Box(c \wedge d)$, thus we proved that $(c \wedge d) \in out_2(Q, \{a \vee b\})$.



IO_Logic_out2.thy (~/Isabelle/)

```
(*Q={(a,c), (a,d), (b, c∧d) } A = {a ∨ b} *)
definition "Q ≡  (λX. X=(a,c) ∨ X=(a,d) ∨ X=(b,c ∧ d))" declare Q_def[Defs]
(*(c ∧ d) ∈ out2(Q,A) *)
(* c ∧ d ∈ Q(L) *)
lemma "G_L Q (c ∧ d)"
  by (meson G_L_def Q_def)
(* c ∧ d ∈ Cn(Q(L)) →  Sledgehammer times out / Nitpick out of scope *)
lemma " (out2 Q (c ∧ d))" nitpick[user_axioms,show_all] oops
(* Verify modal logic part *)
lemma "⌊ ((a ⊃ □c) ∧ (a ⊃ □d) ∧ (b ⊃ □(c ∧ d)) ∧ (a ∨ b)) ⊃ □(c ∧ d) ⌋"
  using kand_def kbox_def kimp_def kor_def kvalid_def by auto
(* Both parts together with the old formulation → Time out *)
lemma "⌊ ((a ⊃ □c) ∧ (a ⊃ □d) ∧ (b ⊃ □(c ∧ d)) ∧ (a ∨ b)) ⊃ □(c ∧ d) ⌋ ∧ (out2 Q (c ∧ d))"
  oops
(* New formulation for c ∧ d ∈ Cn(Q(L)) *)
lemma "⌊(c ∧ d ∧ (c ∧ d))⊃(c ∧ d) ⌋" by (simp add: kand_def kimp_def kvalid_def)
(* Both parts together with new formulation → Proof found *)
lemma "⌊ ((a ⊃ □c) ∧ (a ⊃ □d) ∧ (b ⊃ □(c ∧ d)) ∧ (a ∨ b)) ⊃ □(c ∧ d) ⌋
         ∧ ⌊(c ∧ d ∧ (c ∧ d))⊃(c ∧ d) ⌋ "
  using kand_def kbox_def kimp_def kor_def kvalid_def by auto
```

Proof state [✓] Auto update [✓] Update Search: 100%

```
theorem ⌊(a ⊃ □ c) ∧ (b ∨ e ⊃ □ c) ∧ (a ∨ b) ⊃ □ c⌋ ∧ ⌊c ∧ c ⊃ c⌋
```

```
158,1 (5477/10823)          Input/output complete          (isabelle,isabelle,UTF-8-Isabelle)Nmro UG 200/748MB 1:55 PM
```

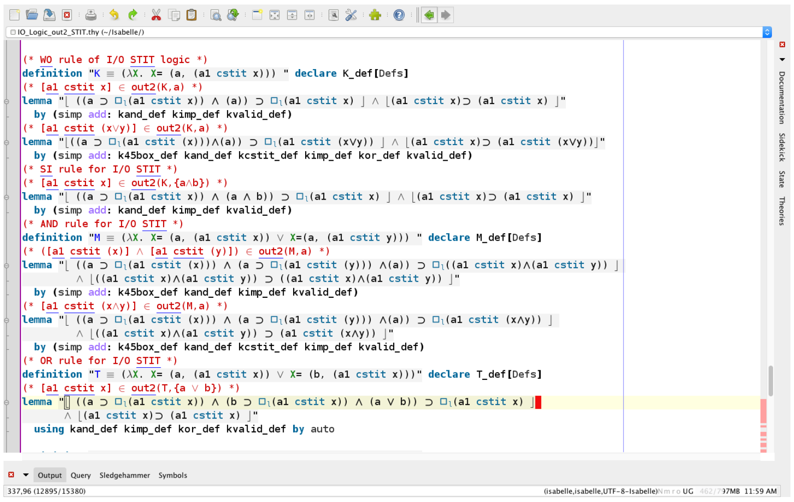
Figure 3.11: Isabelle/HOL: out_2 example

3.6.3 Applying out_2 to formulas of T-STIT logic

The language of T-STIT logic already consists of an operator $\Box$. It satisfies all principles of a modal logic **S5**. Also, the modal formulation of out_2 makes use of an operator $\Box$, but this one satisfies different principles, namely the ones from a modal logic **K45**. So when combining both logics in *Isabelle/HOL*, we have to clearly distinguish them. Therefore we let $\Box$ denote the lifted $\Box$ operator for $\mathcal{L}_{\text{T-STIT}}$ and $\Box_l$ denotes the lifted $\Box$ operator for the modal logic **K45**.

In order to verify the driving example for the out_1, it was necessary to implement a corresponding model into *Isabelle/HOL* in order for $Sledgehammer$ to come up with proofs. However, for the out_2 operation in *Isabelle/HOL*, a model does not have to be provided. It is sufficient to define the set of conditional norms G, namely $G = \{(b, [\alpha]\neg a), (\neg b, [\alpha]a)\}$, and we can check the same statements also hold for the out_2 operation. And indeed, the statements, which did hold for out_1, held as well for out_2.

We can also illustrate the soundness of the logic on some examples in *Isabelle/HOL*. See figure 3.12. Take for instance $K = \{(a, [\alpha]x)\}$. Obviously we have that $[\alpha]x \in out_2(K, a)$. By *weakening the output (WO)*, we have that $[\alpha](x \vee y)$ and x are in $out_2(K, a)$ since $[\alpha](x \vee y)$ and x are logical consequences of $[\alpha]x$. Further we can conclude that $[\alpha]x \in out_2(K, \{a \wedge b\})$. This is guaranteed by *strengthening the input (SI)* since we have that $[\alpha]x \in out_2(K, a)$ and a logically follows from $a \wedge b$. For a set $M = \{(a, [\alpha]x), (a, [\alpha]y)\}$, we have that $[\alpha]x$ and $[\alpha]y$ are in $out_2(M, a)$. By the *conjunction of the output (AND)*, we get that $([\alpha]x \wedge [\alpha]y) \in out_2(M, a)$ and since $([\alpha]x \wedge [\alpha]y) \supset [\alpha](x \wedge y)$, we also get that $[\alpha](x \wedge y) \in out_2(M, a)$. Finally, for a set $T = \{(a, [\alpha]x), (b, [\alpha]x)\}$, we obtain that $[\alpha]x \in out_2(T, a)$ and $[\alpha]x \in out_2(T, b)$ which implies by the *OR* rule that $[\alpha]x \in out_2(T, a \vee b)$.



```
(* WO rule of I/O STIT logic *)
definition "K ≡ (λX. X= (a, (a1 cstit x))) " declare K_def[Defs]
(* [a1 cstit x] ∈ out2(K,a) *)
lemma "⌊ ((a ⊃ □ₗ(a1 cstit x)) ∧ (a)) ⊃ □ₗ(a1 cstit x) ⌋ ∧ ⌊(a1 cstit x)⊃ (a1 cstit x) ⌋"
  by (simp add: kand_def kimp_def kvalid_def)
(* [a1 cstit (x∨y)] ∈ out2(K,a) *)
lemma "⌊((a ⊃ □ₗ(a1 cstit (x)))∧(a)) ⊃ □ₗ(a1 cstit (x∨y)) ⌋ ∧ ⌊(a1 cstit x)⊃ (a1 cstit (x∨y))⌋"
  by (simp add: k45box_def kand_def kcstit_def kimp_def kor_def kvalid_def)
(* SI rule for I/O STIT *)
(* [a1 cstit x] ∈ out2(K,{a∧b}) *)
lemma "⌊ ((a ⊃ □ₗ(a1 cstit x)) ∧ (a ∧ b)) ⊃ □ₗ(a1 cstit x) ⌋ ∧ ⌊(a1 cstit x)⊃ (a1 cstit x) ⌋"
  by (simp add: kand_def kimp_def kvalid_def)
(* AND rule for I/O STIT *)
definition "M ≡ (λX. X= (a, (a1 cstit x)) ∨ X=(a, (a1 cstit y))) " declare M_def[Defs]
(* ([a1 cstit (x)] ∧ [a1 cstit (y)]) ∈ out2(M,a) *)
lemma "⌊ ((a ⊃ □ₗ(a1 cstit (x))) ∧ (a ⊃ □ₗ(a1 cstit (y))) ∧(a)) ⊃ □ₗ((a1 cstit x)∧(a1 cstit y)) ⌋
         ∧ ⌊((a1 cstit x)∧(a1 cstit y)) ⊃ ((a1 cstit x)∧(a1 cstit y)) ⌋"
  by (simp add: k45box_def kand_def kimp_def kvalid_def)
(* [a1 cstit (x∧y)] ∈ out2(M,a) *)
lemma "⌊ ((a ⊃ □ₗ(a1 cstit (x))) ∧ (a ⊃ □ₗ(a1 cstit (y))) ∧(a)) ⊃ □ₗ(a1 cstit (x∧y)) ⌋
         ∧ ⌊((a1 cstit x)∧(a1 cstit y)) ⊃ (a1 cstit (x∧y)) ⌋"
  by (simp add: k45box_def kand_def kcstit_def kimp_def kvalid_def)
(* OR rule for I/O STIT *)
definition "T ≡ (λX. X= (a, (a1 cstit x)) ∨ X= (b, (a1 cstit x)))" declare T_def[Defs]
(* [a1 cstit x] ∈ out2(T,{a ∨ b}) *)
lemma "⌊ ((a ⊃ □ₗ(a1 cstit x)) ∧ (b ⊃ □ₗ(a1 cstit x)) ∧ (a ∨ b)) ⊃ □ₗ(a1 cstit x) ⌋
        ∧ ⌊(a1 cstit x)⊃ (a1 cstit x) ⌋"
  using kand_def kimp_def kor_def kvalid_def by auto
```

Figure 3.12: Isabelle/HOL: Proof theory of out_2 on examples

Besides those rules which hold for the out_2, we also tried some other rules which do not hold for this operation. For instance, we showed that the rules of *cumulative transitivity* and *transitivity* don't hold for an example when using the out_2 operator. For the former we considered a set of norms $T1 = \{([\alpha]a, [\alpha]x), ([\alpha]a \wedge [\alpha]x, [\alpha]y)\}$ and $Nitpick$ generated a counter-model to show that $[\alpha]y \notin out_2(T1, [\alpha]a)$. For the later, when $T2 = \{([\alpha]x, [\alpha]y), ([\alpha]y, [\alpha]z)\}$ then $[\alpha]z \notin out_2(T2, [\alpha]x)$.

Another interesting aspect of combining STIT with I/O logic is to study the relation between obligations of multiple agents and the obligation of a group of agents. For example, let's say that in a case of an emergency in a hospital, it is obligatory for person 1 to lock all the windows whereas person 1 is obligated to open all the security doors. So in an emergency scenario, it is then obligatory for the group, which consists of person 1 and 2, to lock all the windows and open all the security doors. So let the statement a denote the case of emergency. The statements x and y express closing the windows and opening the security doors, respectively. The two agents are denoted by α and β, and refer to person 1 and 2, respectively. Agt will denote the group of agents which consists of agent α and β. Then we can define the set of conditional norms as follows $S = \{(a, [\alpha]x), (a, [\beta]y)\}$, use a as the input fact and we would like to conclude that $[Agt](x \wedge y) \in out_2(S, a)$. When formulating this example in *Isabelle/HOL*, it could indeed be proven that $[Agt](x \wedge y) \in out_2(S, a)$.

3.6.4 List of examples

We evaluated the out_1 and out_2 on as many different examples as possible in order to verify that the operations work as intended. Below is a table containing all the examples to which the operators have been applied. The first column represents the set of conditional norms whereas the second one shows the input set that we used. The third column contains the output that we requested for the example. The fourth and fifth columns tell us whether the requested output is in $out_1(N, A)$ and $out_2(N, A)$, respectively, or not. Thus Yes indicates that the requested output should be detached by the respective output operator under the given input set A and the set of conditional norms N. A No indicates then the opposite. If the cell contains Yes and the color of that cell is green, it indicates that $Sledgehammer$ was able to prove the statement. A red background indicates that $Sledgehammer$ timed out and thus could not provide a proof. A yellow background indicates that $Sledgehammer$ was only able to provide a proof when a certain model was specified. If the cell contains No, we have to disprove the statement by using the tool $Nitpick$. Whenever a counter model was needed, $Nitpick$ was able to generate one and therefore the background color of those cells are always green. The results from the examples for operator out_2 are quite promising. Every statement that contained an out_2 could correctly be proved or disproved by $Sledgehammer$ resp. $Nitpick$. The reason behind this can surely be traced back to the modal formulation of out_2.

N	A	Output(s)	In $out_1(N,A)$?	In $out_2(N,A)$?
$\{(a,x),(b,x)\}$	$\{a\}$	x	Yes	Yes
$\{(a,x),(b,x)\}$	$\{a\}$	$x \lor b$	Yes	Yes
$\{(a,x),(b,x)\}$	$\{a \lor b\}$	x	No	Yes
$\{(a,x),(b,x)\}$	$\{a \land b\}$	x	Yes	Yes
$\{(a,b),(a,c)\}$	$\{a\}$	$b, c, b \land c$	Yes	Yes
$\{(a,a),(a,b),(a,c)\}$	$\{a \land \neg c\}$	$a, a \land b \land c$	Yes	Yes
$\{(a,c),(b,c),(d,c)\}$	$\{a \lor b \lor d\}$	c	No	Yes
$\{(a,c),(b,c),(d,c \land e)\}$	$\{a \lor b \lor d\}$	c	No	Yes
$\{(a,c),(b,c),(d,c \land e)\}$	$\{a \lor b \lor d\}$	$c \land e$	No	No
$\{(a,c),(b \lor e,c)\}$	$\{a \lor b\}$	c	No	Yes
$\{(a,c),(a,d),(b,c \land d)\}$	$\{a \lor b\}$	$c, d, c \land d$	No	Yes
$\{(a,c),(a,d),(b,c \land d)\}$	$\{a\}$	$c \land d$	Yes	Yes
$\{(a,x)\}$	$\{a \land b\}$	x	Yes	Yes
$\{(a,x)\}$	$\{a\}$	$x \lor y$	Yes	Yes
$\{(a,x),(a \land b,d)\}$	$\{a \land b\}$	x	Yes	Yes
$\{(\top,k),(\neg k,a)\}$	$\{\neg k\}$	$k, k \land a$	Yes	Yes
$\{(a,[\alpha]x)\}$	$\{a\}$	$[\alpha]x, \Diamond[\alpha]x, x$	Yes	Yes
$\{(a,[\alpha]x)\}$	$\{a\}$	$[\alpha](x \lor y)$	Yes	Yes
$\{(a,[\alpha]x)\}$	$\{a \land b\}$	$[\alpha]x$	Yes	Yes
$\{(a,[\alpha]x),(a,[\alpha]y)\}$	$\{a\}$	$[\alpha](x) \land [\alpha](y)$	Yes	Yes
$\{(a,[\alpha]x),(a,[\alpha]y)\}$	$\{a\}$	$[\alpha](x \land y)$	Yes	Yes
$\{(a,[\alpha]x),(b,[\alpha]x)\}$	$\{a \lor b\}$	$[\alpha]x$	No	Yes
$\{(a,[\alpha]x),(b,[\alpha]x)\}$	$\{[\alpha]x\}$	$[\alpha]x$	No	No
$\{([\alpha]a,[\alpha]x),([\alpha](a) \land [\alpha](x),[\alpha]y)\}$	$\{[\alpha]a\}$	$[\alpha]y$	No	No
$\{([\alpha]x,[\alpha]y),([\alpha]y,[\alpha]z)\}$	$\{[\alpha]x\}$	$[\alpha]z$	No	No
$\{(a,[\alpha]x),(a,[\beta]y)\}$	$\{a\}$	$[Agt](x \land y)$	Yes	Yes
$\{(a,[Agt]z)\}$	$\{a\}$	$[\alpha](z) \land [\beta](z)$	No	No
$\{(b,[\alpha](\neg a)),(\neg b,[\alpha]a)\}$	$\{b\}$	$[\alpha](\neg a), \neg a$	Yes	Yes
$\{(b,[\alpha](\neg a)),(\neg b,[\alpha]a)\}$	$\{\neg b\}$	$[\alpha](a), a$	Yes	Yes
$\{(b,[\alpha](\neg a)),(\neg b,[\alpha]a)\}$	$\{b\}$	$[\alpha](a)$	No	No
$\{(b,[\alpha](\neg a)),(\neg b,[\alpha]a)\}$	$\{\top\}$	$[\alpha](a)$	No	No
$\{(a \lor b,[\alpha]e)\}$	$\{[\beta]b\}$	$[\alpha](e \lor f)$	Yes	Yes
$\{(a,[\alpha]e),(a,[\beta]f),(b,e \land f)\}$	$\{a\}$	$[\alpha]e, e, e \land f$	Yes	Yes
$\{(a,[\alpha]e),(a,[\beta]f),(b,e \land f)\}$	$\{[\beta]b\}$	$e \land f$	Yes	Yes
$\{(\top,[\alpha]e)\}$	$\{\top\}$	$[\alpha](e \lor f)$	Yes	Yes
$\{(\top,[\alpha\, dstit:\ e])\}$	$\{\top\}$	$[\alpha\, dstit:\ (e \lor f)]$	No	No
$\{(a,\Box x)\}$	$\{a\}$	$\Box x, \Diamond x, x, x \lor y$	Yes	Yes
$\{(a,\Box x)\}$	$\{a\}$	$[\alpha]x, [\beta]x, [Agt]x$	Yes	Yes
$\{(a,\Box x)\}$	$\{a\}$	$[\alpha]x \land [\beta]x$	Yes	Yes

4 | Moral Luck

This chapter presents the phenomenon of moral luck. Before identify which aspects of moral luck can actually be studied by our logic and how they are related, the chapter documents the four different kinds of moral luck identified by Thomas Nagel [25]. Finally we use examples from moral luck as test cases for our logic.

4.1 Introduction to moral luck

The literature of moral luck is focused around the question whether luck can ever make a moral difference or not. Chance affects our lives in much more ways that one might actually think. However, one might consider morality as an area where luck does not play such a powerful role. The following example actually illustrates though how much of an impact luck can have when dealing with moral judgments.

Example 7 (Drunk drivers) Let A and B be next door neighbors. At one evening, both go to the same party and get equally drunk. When the party comes to an end, A and B get to their own vehicles and since they are neighbors, both take the same road in order to drive home. The only difference is that A will leave a few minutes earlier than B. The roads are pretty much deserted at that time and A manages to drive home safely even with a high percentage of alcohol in his blood. A couple of minutes later, B drives down the same road and suddenly a child appears in front of his car. Since B also drank a lot at the party, his reaction time is impaired by the alcohol and it makes it impossible for him to stop and swerve to avoid hitting and killing the child.

The question in this example is as follows, who deserves more blame? In the 20th century, the British philosopher Bernard Williams introduced the term moral luck. It describes scenarios where a moral agent is assigned a moral judgment. The agent gets either praised or blamed for his or her action and its consequences, even if it is clear that either the action or its consequences were beyond the agent's control. Williams and the American philosopher Thomas Nagel developed the subject of moral luck in their respective work [30, 25]. Both of them argue that luck actually can make a moral difference and used similar examples to illustrate the issue when dealing with moral responsibility. In philosophy, moral responsibility refers to acts or states of affairs for which an agent can get praised or blamed.

Back to the example, we might notice at a first look that it seems that person B is clearly more blameworthy due to the fact that B killed the child and A did not. But looking more deeply into the situation, we have that A and B made the equally blameworthy choice and decided to drive home while being intoxicated. However, B encountered an external factor while driving down the road, namely a child. Imagine now, that the child had crossed the path of person A. Being drunk, A would also have been unable to avoid hitting the child.

Both drivers neglected the fact that one should not take the road while having a high percentage of alcohol in the blood. Still, both of them decided to drive home and neither one of them intended to hit and kill anyone. It seems, therefore, irrational that B is more blameworthy than A. Both of them neglected the same thing and only because of luck, nobody got harmed and killed in the scenario of person A. Therefore, we say that A got morally lucky. Such an example illustrates the problem of moral luck. But luck should not affect the degree of moral responsibility.

So a question that one may ask is, how to figure out and know for sure that someone should deserve praise or blame for some situation that just happened?

4.2 Ought implies Can

A well-known ethical formula in moral philosophy is the principle of *Ought implies can*. Among philosophers, this principle is commonly used when dealing with moral obligations. It states that if an agent ought or should perform an action morally, then he or she must be able to perform it. Alternatively, an agent is only morally required to do things that are possible for him or her under natural conditions. According to this principle, obligations of a human are restricted to what is humanly possible and therefore we have a limit on ethical responsibility. As a consequence, some obligations might break due to changes in real-world circumstances. Consider the case where a person A made a promise to person B in order to meet for lunch. Consequently, A has the obligation to keep this promise. On his or her way to meet B, person A, however, ends up in a car crash and gets knocked unconscious. Several minutes later, an ambulance arrives and A gets brought to the nearest hospital. A is now unable to meet B. Even though A promised to have lunch with B, A did nothing wrong in this situation. This may lead some of us to think that we can neglect obligations by intentionally getting ourselves into situations where we are unable to do what we are obligated to do. For instance, the court charges someone to pay child support. Then this person decides to go to a casino and spends every dollar on gambling but he or she loses all of his or her fortune in the end. Having no money at all, the person is unable to pay the child support, he or she has then no obligation to do so. However, this argument is not valid. If person A has an obligation for X, and X depends on doing Y, then if Y falls within the control of A, then person A has an obligation for Y as well. In the case of the previous example, the person has an obligation to pay child support and thus he or she also has an obligation not to visit a casino and waste away all the money.

From a logical point of view, the principle of *Ought implies can* also makes absolute sense. Therefore it belongs to the few principles, which basically everyone agrees. With this principle in mind, if an agent faces a situation, which is out of his or her control, then we should agree on that he or she cannot be morally responsible for it.

Imagine that someone would cut the brake lines of somebody's car without getting noticed. As a consequence, the driver of that car would end up in an accident but he is not morally responsible for the resulting injuries and damage. However, he still took part in the chain of events that resulted in injuries and damage. Had the driver decided not to get into his car, then this accident would never have taken place.

This small example illustrates the difference between casual and moral responsibility. The former states whether agents had been involved in a series of events or not whereas the latest assigns responsibility to the agents. Depending on what happened, the agents receive either positive or negative judgments. However, the concept of moral responsibility is only considered for moral agents. Meaning that we only consider those agents that have the ability to distinguish between right and wrong and then make choices accordingly.

Looking back at the example of the drunk drivers and if we consider that one is only morally responsible for what is in your control, then we have to conclude that both drivers should be equally blameworthy. Alternatively, we say that A and B are morally equal. Neither person is relevantly worse than the other. The fact that the child ran into the road, is something beyond B's control and therefore he has no influence on it.

A moral assessment cannot depend on factors which are out of range of the agent's control. This is statement is captured by the condition of control. In particular, it states that agents are morally assessable only to the extent that what they are assessed for depends on factors under their control. This implies then if the differences in the actions of two agents are only because of factors outside their control than the differences in their actions cannot be relevant for the moral evaluation. Thus the problem of moral responsibility is much more complex than it looked at first sight.

4.3 The four kinds of moral luck

Nagel states that external factors might affect the moral quality of our actions and he goes on by describing these effects in terms of different kinds of luck. In particular, Nagel identified four kinds of moral luck.

4.3.1 Constitutive moral luck

Constitutive moral luck is referred to as the luck that deals with our own constitution or personal character. Thus it is concerned with the individual character traits and motivations of an agent. Nagel claims that some people are born generous whereas some of us are more greedy and envious. On one side, some people tend to get angry very quickly and on the other, there are people who are more easy going and very calm. Therefore everyone has a different temperament.

According to Nagel, people tend to make their choices based on their personality which implies that our own personal traits dictate our actions. As a consequence, different personalities will lead to different choices and we act as who we are. However one may argue that our personality, to some extent, is shaped by influences which are out of our control. Such influences include parental as well as academic education, environmental and genetics.

For example, a person may not like to share his or her snacks with another person. But a lot of other people find sharing food with each other to be second nature. Even though that one person may have gotten used to offering to whoever is around him or her, it is easy for others to judge that person by his or her natural disposition towards not sharing as selfish and greedy. Regardless of people's intentions or actions, they are held praiseworthy or blameworthy for parts of their personality that they do not control.

4.3.2 Circumstantial moral luck

Circumstantial moral luck is concerned with the surrounding or the situation the moral agent finds himself in. According to Nagel, people don't have any control over the circumstances they face during their lifetime. But one should keep in mind that those surrounding play a huge factor when it comes to making a choice.

An example, illustrating the concept of circumstantial moral luck, can be found in Nagel's paper [25]. It deals with the followers and supporters of the Nazi regime in Germany. Some of those people are morally blameworthy for terrible deeds during that time. Others can be morally blamed for allowing those deeds to occur because they did not make the effort to oppose them. However, imagine that those people would have been moved to different countries some years before this regime had been in control, then those people would have possibly lived different lives and would not receive the same amount of moral blame. But this is due to the luck of circumstances in which they are surrounded.

This makes it clear, that people who have done things for which they deserve moral blame, would not have done those if they had found themselves in different circumstances.

4.3.3 Resultant moral luck

Resultant moral luck is the most obvious form of luck. Thus it is concerned with the way things turn out and deals with the consequences of actions and situations. When two people are performing the same action or making the same choice, the degree of responsibility might differ due to subsequent events which are out of their control. Resultant luck may arise in cases of negligence, intentional wrongdoing or decision under uncertainty.

For instance, take two truck drivers who have forgotten to get their brakes checked. Due to this negligence, both drivers experience a moment where their brakes fail, which will then end up in a crash. One driver only caused material damage, but the other one finds a dead pedestrian in front of his truck. In both scenarios, we notice two different outcomes of the crashes. In one situation, a pedestrian appeared and got unluckily hit by the truck whereas, in the other, the pedestrian did not appear or avoided being hit. Both drivers were negligent in the same way, however, in one situation, it ended up in a tragedy. In the other one, it was just a matter of luck that nobody ended up dead. The same kind of luck was also illustrated by the example of the two drunken drivers.

A successful and unsuccessful murder attempt is considered as a case of intentional wrongdoing. In many countries, the charges for an attempting murder versus actual murder are quite different. But the intention in both cases is exactly the same and the person is judged by the end result of whether someone died or not. This might be something which is outside of the agent's control. For instance, a bird could have been flown into the path of a bullet or maybe the victim have been worn a bullet proofed vest or the trigger of the weapon is not working, resulting in a failed murder attempt. From a moral point of view, murdering someone is wrong. Additionally, the law considers murder as a crime, therefore morality and law overlap in the case of murder. However, in law, the punishment for attempted and actual murder is not the same. The law prioritizes outcomes over intentions meaning that it is primarily concerned with definite outcomes and secondly with intentions. The difference between killing someone and grievous bodily harm is not traced back to the intentions or the inflicted wounds, but it is a matter of the outcome, whether someone has died or not. In contrast, our moral judgment may be identically in the case of murder and vicious assault.

Decisions under uncertainty deal with the issue that an agent takes a highly risky action. Due to that, the result of that action is surely unpredictable, so that under no circumstances one is able to foresee the outcome. But in the end, the moral judgment will be based on that outcome of the agent. For example, take a person who is a merciless and ruthless leader of a revolution and fights violently against an authoritarian regime. It is clear to him that if he fails he is going to be responsible for the anarchy. But he will also be held accountable for the suffering and death of the people, believing in him, that was in vain due. However, a success against the regime would mean that he gets justification for the outcome.

4.3.4 Causal moral luck

Causal moral luck can be seen as a combination of constitutive and circumstantial moral luck. Therefore it is the luck due to antecedent circumstances and according to Nagel, it is largely related to the classical problem of free will.

It is all about the events that have influenced the person that you are. For instance, someone may have read a life-changing book or a person may have been inspired by a teacher during his or her studies. A person's personality is formed by the events that have happened to him or her.

Further, a person is unable to control where he or she comes from. But undeniable, it has a huge impact on what a person is going to become. Some people may have encountered hard struggles during their lives, which made them more viciousness, while others may not have faced such things. People can be raised in the most terrible circumstances but still have amazing personalities. However, it also holds vice versa. The best possible environment does not guarantee good character traits.

Nagel has been criticized for introducing this kind of moral luck as a separate category. For many, it is considered as largely redundant since it does not include any new cases which have not been already separately covered by constitutive or circumstantial luck.

4.4 Relation to our logic

Moral luck is a rather philosophical topic, however, since we want to use examplew from it as test cases, we have to relate it to our presented logic, T-STIT I/O logic, and identify the aspects that can be analyzed by it.

First, we must be able to formalize and model examples from moral luck. Since we presented the different kinds of moral luck, we have to determine which out of the four kinds is suitable for our logic, meaning that the logic is able to model such examples of this kind. Clearly, our logic is not able to capture an agent's personality, character or circumstances. This already rules out three out of the four kinds of moral luck. An agent's actions and its consequences are covered in resultant moral luck. With T-STIT logic, we are able to capture the concepts of agency and action, which means then that we put our focus on examples from resultant moral luck and T-STIT logic is used to formalize the statements to model a scenario of an agent in the example.

In I/O logic, we have an input set A which describes our state of affairs and if $\varphi \in out(N, A)$, we say that φ is obligatory under the norms N for such an input set A. This is how obligations are determined in I/O logic. So when we model an example of moral luck, the set A is used to represent the scenario of an agent in such an example whereas the set N contains the desired norms. As already mentioned before, T-STIT logic is used for the modeling part, thus the sets A and N contain formulas of this logic. With our I/O operators, we are able to determine which obligations hold for the different agents in their respective situations. But which other aspects from moral luck can we cover with deontic logic, in particular, I/O logic? Looking for instance at the example of two drunk drivers, we have that both of them made the choice to drive home under the influence of alcohol. Deontic logic is the study that is concerned with obligations and looking at this example from a deontic logical point of view, we clearly want that it is forbidden for the agents to be drunk and take the road. Alternatively, it is obligatory not to be drunk and not to drive at the same time. However, in both cases, both agents still made the choice not to follow this obligation. By doing so, we say that both of them committed a violation of an obligation. A violation is formally represented as follows: for a set of conditional norms N and an input set A, an obligation for φ has been violated if and only if $\varphi \in out(N, A)$ and $\neg\varphi \in Cn(A)$. Alternatively this can be expressed as $\perp \in out(N, A) \cup Cn(A)$. Violations of obligations can be seen as not permitted actions or even forbidden actions. In an ideal system, we expected that the agents behave according to our norms and obligations. However, in non-ideal systems, agents might actually behave differently, and the conflict between an expected and an actual behavior causes those violations of obligations. An agent is then responsible for a violation of an obligation φ if there has been an obligation of φ but they decided to perform an action which led to $\neg\varphi$.

4.5 Example - Drunk Drivers

4.5.1 Formulation

A particular example of resultant moral luck is the one regarding the two drunk drivers, mentioned at the beginning of this chapter, and we use it as a test case for our logic. We want to investigate which obligations are holding in the cases of the two drivers. With the operators out_1 and out_2, we are able to express obligations for some set of conditional norms and a given input set. STIT logic is used to represent the actions and choices of the agents. To formulate this example, we need two agents α and β which are representing the two drivers. The scenarios of both agents are almost identical, the only difference comes from the fact that one agent ended up killing someone whereas the other one did not. We will focus first on the case for the former agent, which will be denoted by α, and specify an input set to model his scenario. The propositions $Drunk_\alpha$ and $[\alpha]Drive$ represent the facts that agent α got drunk and that he still decided to drive, respectively. In the case of α, we also had that a child appeared on the road, thus we use the proposition $Jump$ as a corresponding representation. The influence of alcohol impairs the agent's driving behavior in such a way that he is no longer able to drive carefully. In STIT logic, the formula $\Diamond[\alpha]\varphi$ denotes that agent α has the ability to see to it that φ. Thus, we can represented this sentence by the proposition $Drunk_\alpha \to \neg\Diamond[\alpha]DriveCarefully$. Finally, we have that if the agent decides to drive, but not being able to do it carefully, and there is somebody appearing on the road, then this results in killing or hurting that person. Thus the last input fact is formalized as $(\neg\Diamond[\alpha]DriveCarefully \wedge [\alpha]Drive \wedge Jump) \to (Kill \vee Hurt)$. We will use A to denote input set, containing all those facts of agent α. In addition to the input facts, we use N to denote the set of conditional norms containing the following identified conditional norms:

- $(\top, \neg Kill \wedge \neg Hurt)$
  This norm states that in any situation, it is obligatory that one does not kill and does not hurt somebody else. Alternatively, under no circumstances, it is not allowed to kill or hurt somebody.

- $(\top, \Box DriveCarefully)$
  This norm states that in any situation, it is obligatory that it is necessary to drive carefully. One is obligated to always drive carefully.

– $(\neg\Diamond[\alpha]DriveCarefully, [\alpha]Stay)$
  This norms that under the condition that agent α is not able to drive carefully, it is obligatory that agent α sees to it that he stays at his current place. By staying at the current place, it indicates that agent α should not take the road.

For agent β, we have more or less the same input set A' containing the following propositions: $Drunk_\beta$, $[\beta]Drive$, $\neg Jump$, $Drunk_\beta \to \neg\Diamond[\beta]DriveCarefully$ and $(\neg\Diamond[\beta]DriveCarefully \wedge [\beta]Drive \wedge Jump) \to (Kill \vee Hurt)$. For the set of conditional norms N', the first two norms are identical to ones we used for agent α. The third norm will just be adapted for agent β, meaning that we have $(\neg\Diamond[\beta]DriveCarefully, [\beta]Stay) \in N'$.

4.5.2 Verification

In the case of agent α, when applying the operators out_1 to the sets N and A, we get the following output set:

$$out_1(N, A) = Cn(\{\neg Kill \wedge \neg Hurt, \Box DriveCarefully, [\alpha]Stay\})$$

and therefore we obviously get that $(\neg Kill \wedge \neg Hurt) \in out_1(N, A)$ and $[\alpha]Stay \in out_1(N, A)$. Further, since $\Box DriveCarefully \to [\alpha]DriveCarefully$ and $[\alpha]DriveCarefully \to \Diamond[\alpha]DriveCarefully$, we have $\Diamond[\alpha]DriveCarefully \in out_1(N, A)$. All of these statements are also part of $out_2(N, A)$. The operators out_1 and out_2 behaved as expected. Also when formulating this example in Isabelle/HOL for the out_2 operator, we are able to obtain proofs that those propositions are included in $out_2(N, A)$. But looking more closely at the example, we notice the following. The output set is consistent, thus $\bot \notin out_i(N, A)$ for $i \in \{1, 2\}$. However, looking back at the input set A, we notice that $Kill \vee Hurt \in Cn(A)$ and $\neg\Diamond[\alpha]DriveCarefully \in Cn(A)$. The agent α is not able to drive carefully even though there is an obligation which tells us that is necessary to drive carefully and therefore we have a conflict between those two and we say that this obligation has been *violated*. In particular, we have two violations:

– $\Diamond[\alpha]DriveCarefully \in out_1(N, A)$ but $\neg\Diamond[\alpha]DriveCarefully \in Cn(A)$.

– $(\neg Kill \wedge \neg Hurt) \in out_1(N, A)$ but $(Kill \vee Hurt) \in Cn(A)$.

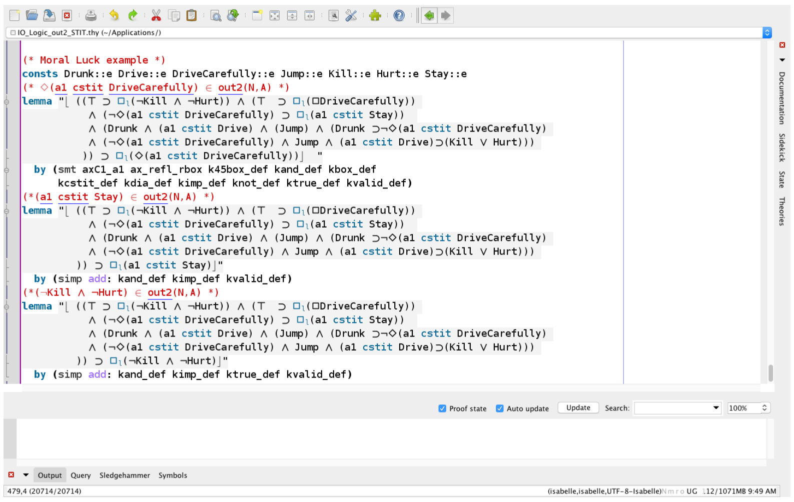
By applying the out_1 operator to the sets N' and A', we get a similar output set for agent β.

$$out_1(N', A') = Cn(\{\neg Kill \wedge \neg Hurt, \Box DriveCarefully, [\beta]Stay\})$$

Since $(Kill \vee Hurt) \notin Cn(A')$, we have that agent β did not violated the obligation of $\neg Kill \wedge \neg Hurt$. However, we still observe a violation:

– $\Diamond[\beta]DriveCarefully \in out_1(N', A')$ but $\neg\Diamond[\beta]DriveCarefully \in Cn(A')$

Due to violations, our input set gets inconsistent with our output. This implies also that it is not possible to satisfy all the norms in the given situation. Dealing with violations has always been a challenge in traditional I/O logic. Additionally, Contrary-To-Duty (CTD) reasoning and deontic dilemmas (unsolvable conflicts between obligations) are difficult to handle for the standard I/O operators out_1 and out_2. This example can be seen as CTD reasoning. The norm $(\top, \Box DriveCarefully)$ represents a primary obligation whereas $(\neg\Diamond[\alpha]DriveCarefully, [\alpha]Stay)$ is the Contrary-To-Duty obligation to it and tells us what is obligatory in the case when our primary obligation has been violated.



```
(* Moral Luck example *)
consts Drunk::e Drive::e DriveCarefully::e Jump::e Kill::e Hurt::e Stay::e
(* ◇(a1 cstit DriveCarefully) ∈ out2(N,A) *)
lemma "⌊ ((⊤ ⊃ □ₗ(¬Kill ∧ ¬Hurt)) ∧ (⊤  ⊃ □ₗ(□DriveCarefully))
          ∧ (¬◇(a1 cstit DriveCarefully) ⊃ □ₗ(a1 cstit Stay))
          ∧ (Drunk ∧ (a1 cstit Drive) ∧ (Jump) ∧ (Drunk ⊃¬◇(a1 cstit DriveCarefully)
          ∧ (¬◇(a1 cstit DriveCarefully) ∧ Jump ∧ (a1 cstit Drive)⊃(Kill ∨ Hurt)))
         )) ⊃ □ₗ(◇(a1 cstit DriveCarefully))⌋  "
  by (smt axC1_a1 ax_refl_rbox k45box_def kand_def kbox_def
      kcstit_def kdia_def kimp_def knot_def ktrue_def kvalid_def)
(*(a1 cstit Stay) ∈ out2(N,A) *)
lemma "⌊ ((⊤ ⊃ □ₗ(¬Kill ∧ ¬Hurt)) ∧ (⊤  ⊃ □ₗ(□DriveCarefully))
          ∧ (¬◇(a1 cstit DriveCarefully) ⊃ □ₗ(a1 cstit Stay))
          ∧ (Drunk ∧ (a1 cstit Drive) ∧ (Jump) ∧ (Drunk ⊃¬◇(a1 cstit DriveCarefully)
          ∧ (¬◇(a1 cstit DriveCarefully) ∧ Jump ∧ (a1 cstit Drive)⊃(Kill ∨ Hurt)))
        )) ⊃ □ₗ(a1 cstit Stay)⌋"
  by (simp add: kand_def kimp_def kvalid_def)
(*(¬Kill ∧ ¬Hurt) ∈ out2(N,A) *)
lemma "⌊ ((⊤ ⊃ □ₗ(¬Kill ∧ ¬Hurt)) ∧ (⊤  ⊃ □ₗ(□DriveCarefully))
          ∧ (¬◇(a1 cstit DriveCarefully) ⊃ □ₗ(a1 cstit Stay))
          ∧ (Drunk ∧ (a1 cstit Drive) ∧ (Jump) ∧ (Drunk ⊃¬◇(a1 cstit DriveCarefully)
          ∧ (¬◇(a1 cstit DriveCarefully) ∧ Jump ∧ (a1 cstit Drive)⊃(Kill ∨ Hurt)))
        )) ⊃ □ₗ(¬Kill ∧ ¬Hurt)⌋"
  by (simp add: kand_def kimp_def ktrue_def kvalid_def)
```

Figure 4.1: Isabelle/HOL: Moral luck example

4.6 Example - Murder attempt

4.6.1 Formulation

Another example of resultant moral luck is the earlier mentioned case of a successful and unsuccessful murder attempt. In one scenario, an assassin shoots and kills his target whereas, in the other one, the assassin is not able to shoot and kill the target because of a malfunctioning trigger of his weapon. To formalize this example, we need to consider again agents α and β which are denoting the two assassins. Next, we need to define some propositions for the input sets of the agents to model their scenarios. Agent α is the assassin in the first scenario and the input set A contains the following propositions:

- $[\alpha]Pull$
- $TriggerWorking$
- $([\alpha]Pull \wedge TriggerWorking) \rightarrow Kill$

The input set A' for agent β, the assassin in the second scenario, is almost identically but in the case for β, the trigger is not working properly.

- $[\beta]Pull$
- $\neg TriggerWorking$
- $([\beta]Pull \wedge TriggerWorking) \rightarrow Kill$

The propositions $[\alpha]Pull$ and $[\beta]Pull$ are representing the facts that α and β chose to pull the trigger of their weapons, respectively. The proposition $Trigger$ $Working$ is representing the fact that the trigger of the weapon is working properly. If an agent decides to pull the trigger and this trigger is working as intended then this results in shooting and killing somebody with his weapon. This is captured by the propositions $([\alpha]Pull \wedge TriggerWorking) \rightarrow Kill$ and $([\beta]Pull \wedge TriggerWorking) \rightarrow Kill$. Finally, the set of conditional norm N will contain the following norm:

- $(\top, \neg Kill)$
  In a similar way as in the first example, this norms states that in any situation, it is obligatory that one does not kill somebody else.

4.6.2 Verification

Having defined the input and the conditional norms sets, we can apply the operator out_1 to the sets. For both agents, we obviously have that:

$$out_1(N, A) = out_1(N, A') = Cn(\{\neg Kill\})$$

For agent β, we detached the obligation of not killing somebody. Since $Kill \notin Cn(A')$, agent β did not kill someone, hence he also did not violate this obligation. However, we observe again a violation of a norm in the case of agent α. We have that $\neg Kill \in out_1(N, A)$ but $Kill \in Cn(A)$. This results also in an inconsistency between the output and the input set and shows again the weakness of traditional I/O logic.

4.7 Proposed solutions

A first solution to tackle this problem is to make use of constraints in I/O logic. This is referred to as *Constraint I/O logic (cIOL)* [24]. It introduces a set C, the so-called set of constraints, as an extra parameter and is used to filter out excessive output. Next, we have the following two definitions [24] :

- $maxfamily(N, A, C)$ is the set of $\subseteq$-maximal subsets N' of N such that $out(N', A)$ is consistent with C.
- $outfamily(N, A, C) = \{out(N', A) \mid N' \in maxfamily(N, A, C)\}$ where $out = out_i$ with $i \in \{1, 2, 3, 4\}$

The constrained output operator out_c is then defined as follows [24]:

$$out_c(N, A) = \cup / \cap\, outfamily(N, A, C)$$

Using cIOL, we can actually handle for instance the example of the drunk drivers without getting inconsistency between the output and the input set. Of course the results depend on the choice for the set of constraints C. Set $C = \varnothing$ and $out = out_1$, then $maxfamily(N, A, \varnothing) = \{N\}$ and $outfamily(N, A, \varnothing) = \{out(N, A)\}$. Then $out_c(N, A) = out(N, A)$ and we get the same solution as we did before.

Set $C = A$, then $maxfamily(N, A, A) = \{\{(\neg\Diamond[\alpha]DriveCarefully, [\alpha]Stay)\}\}$ and $outfamily(N, A, A) = \{Cn([\alpha]Stay)\}$. Thus $out_c(N, A) = Cn([\alpha]Stay)$ and we have consistent output which is also consistent with our input. Since the agent violated two obligations in the situation, it is therefore clear that they no longer hold in that case. Being able to handle such violations is an advantage of cIOL over the traditional unconstrained I/O logic. However, cIOL has to drawback of having no proof theory.

Another way to handle this example is to consider the I/O logic defined in the following work [27]. It presents an unconstrained I/O logic with a consistency check that still preserves a proof theory. In [27], the output operator is denoted by O and in terms of semantics, the authors propose that $x \in O(N, A)$ if and only if there is a finite set $M \subseteq N$ and a set $B \subseteq Cn(A)$ such that $M(B) \neq \varnothing$ and

i. $x \dashv\vdash \wedge M(B)$ where $x \dashv\vdash y$ stands short for $x \vdash y$ and $y \vdash x$.

ii. $\forall (a, x) \in M$, we have that $\{a, x\} \cup B$ is consistent.

On the side of the proof theory, for some conjunction a of elements of A, we say that $(a,x) \in D(N)$ if and only if (a,x) is derivable from N by using the following rules [27]:

- (SI) Strengthening the input: from (a,x) to (b,x) whenever we have $b \vdash a$
- (EQ) Equivalence of the output: from (a,x) to (a,y) whenever we have $x \dashv\vdash y$
- (R-AND) Restricted AND: from (a,x) and (a,y) to $(a,x \wedge y)$ whenever $a \wedge x$ consistent and $a \wedge y$ consistent

To check that $[\alpha]Stay$ is outputted in context of $Cn(A)$, we have to consider $M = \{(\neg\Diamond[\alpha]DriveCarefully, [\alpha]Stay)\}$ and $B = Cn(A)$. Then $M(B) = \{[\alpha]Stay\}$ and $\wedge M(B) \dashv\vdash [\alpha]Stay$. Moreover, $\{\neg\Diamond[\alpha]DriveCarefully, [\alpha]Stay\} \cup B$ is consistent, thus $[\alpha]Stay \in O(N,A)$.

As long as the set B contains the proposition $\neg\Diamond[\alpha]DriveCarefully$, the proposition $\Box DriveCarefully$ is not outputted due to the consistency check. To output this proposition, M needs to contain at least the element $\{(\top, \Box DriveCarefully)\}$ and we choose $B = Cn(A)$ to keep the context close to the intended input. However, this will never pass the consistency test and thus $\Box DriveCarefully$ is not outputted. The proposition $\neg Kill \wedge \neg Hurt$ is also never part of the output as long as B is equal to A or $Cn(A)$. Given that the context is $Cn(A)$ then the agent α is not able to drive carefully, the obligation that agent α sees to it that he stays becomes operative whereas the obligation to always drive carefully is inactive. The other obligation of not killing and not hurting somebody is also out since in the context of $Cn(A)$, agent α ended up killing or hurting somebody.

However, it is still possible to output the propositions $\neg Kill \wedge \neg Hurt$ and $\Box DriveCarefully$. Since $B \subseteq Cn(A)$, we can choose $B = \{\top\}$, $M = N$ and obviously we will have the conjunction of those mentioned propositions in the output. However, by changing B to such a set, we are alternating the situation so that it does not correspond anymore to the situation that we had in mind for our input.

For β, we have that $[\beta]Stay \in O(N', A')$ by setting $M = \{(\neg\Diamond[\beta]DriveCarefully, [\beta]Stay)\}$ and $B = Cn(A')$. In contrast to the input set for agent α, we have that $(Kill \vee Hurt) \notin Cn(A')$, and therefore it is not possible to violate the norm $(\top, \neg Kill \wedge \neg Hurt)$. By choosing $M = \{(\top, \neg Kill \wedge \neg Hurt)\}$ and keeping $B = Cn(A')$, we get $(\neg Kill \wedge \neg Hurt) \in O(N', A')$.

With the proposed formalization of the example, we tried to make it as precise as possible. Depending on how one decides to formulate and model such an example, different outcomes might be achieved. Of course, there may exists alternatives when it comes to the formalization for this kind of examples. It is always a subjective matter and it can be a challenging task to tell that a certain formulation of an example is absolutely correct. However, with every formulation, one should capture the fact that both agents have been neglecting an obligation which leads then to the conflict between an output and the input set. Such violations expose the limitations of the traditional I/O operators and have led to development of cIOL and I/O logic with consistency check.

5 | Conclusion

5.1 Summary

The aim of the thesis was to further study the concepts of deontic and agency with regard to automatic reasoning tools. In particular, we were interested in T-STIT logic combined with I/O logic, which we called T-STIT I/O logic, and we used the proof assistant tool *Isabelle/HOL* to verify examples of this logic. In order to do so, we first presented a semantical embedding of T-STIT logic in HOL. Next, we showed the already existing embedding of the out_1 operator and we applied it to formulas of T-STIT logic in *Isabelle/HOL*. Then we extended the work of I/O logic in HOL by introducing an embedding for the out_2 operator in HOL. The embedding has been implemented into the framework *Isabelle/HOL* so that we could apply the out_2 operator first to formulas of propositional logic and afterward to formulas of T-STIT logic. Finally, we related our logic to moral luck, identified which aspects of it are possible to study by the logic and we used examples from it as test cases.

Next, we will give answers to our proposed research questions, identified in chapter 1. The first research question, we wanted to investigate in this thesis, was the following:

RQ1: Due to the recently notable success of embeddings of non-classical logics in HOL and their verifications in *Isabelle/HOL*, is it possible to realize an embedding of STIT logic in HOL or are there any limitations?

In order to realize a semantical embedding of a STIT logic in HOL, we decided to use Lorini's temporal STIT logic [22]. Due to the logic's possible world semantics, the techniques from previous works [1] of the literature could be reused. The embedding was then implemented into *Isabelle/HOL*, a proof assistant for higher-order logic. However, the implementation has its limits. The constraints imposed on the models and the seriality condition of the R_G relation made every model infinite which causes issues for the framework's tools such as *Sledgehammer* and *Nitpick*. To overcome this, we decided to remove the property of seriality from R_G. As a consequence, this step makes the models finite and most of the axioms from Lorini's work [22] could still be proved by the tools. However other axioms were no longer provable because of the removing of the property for the relation R_G. We also tried to prove some laws from Horty's book [19] within *Isabelle/HOL*.

Most of them could also be correctly verified or disproved. As far as we know, this has been the first attempt of an embedding of a modal logic of action in HOL but maybe when the embedding is based on a different semantics, for instance choosing the original semantics of STIT theory, we might be able to overcome the encountered limitations.

In this work, we considered the out_1 and out_2 operators for I/O logic. To verify the out_2 operator on examples in *Isabelle/HOL*, we need a representation for it in HOL. This was captured by our second research question:

RQ2: So far, the literature only documents an embedding of the out_1 operator in HOL [7]. Can the work of I/O logic in HOL be extended by providing a semantical embedding of the out_2 operator?

In contrast to the embedding of the out_1 in HOL, we did not decide to use the traditional formulation of the out_2 for our semantical embedding in HOL but instead, we used its translation into modal logic. By choosing this formulation, the embedding for out_2 could be realized by reusing the same technique already used for other embeddings of various modal logics. Due to its translation into modal logic, the statements were much easier to prove for the tools and the timeout issue, from which the out_1 operators suffers, could be avoided. After adapting the formulation of $x \in Cn(N(\mathcal{L}))$ in *Isabelle/HOL*, all the statements from the examples could be correctly proved or disproved by the corresponding tools. We conclude that the implementation of the out_2 operator is able to deal with more complex statements and the tools can prove them accordingly. This certainly can be traced back to the modal formulation of the out_2, used for the realization of the embedding in HOL.

The two embeddings of those logics are then combined. This allows us then to investigate examples of T-STIT I/O logic in *Isabelle/HOL*. The third research question is concerned about this aspect.

RQ3: Typically, I/O logic is used with propositional logic. How do the I/O operators out_1 and out_2 perform when using STIT logic as the base logic?

Benzmüller and Parent [7] already introduced a semantical embedding of the out_1 in HOL. However when changing the base logic from propositional logic to temporal STIT logic, a lot of statements could not be proven in *Isabelle/HOL* due to the large search space for the *Sledgehammer* tool. By specifying a model, like in the case for the driving model, the scope of the search space gets reduced which allowed then to prove the desired statements of the example. In contrast, the out_2 operator performs far better with T-STIT logic as its base logic than the

out_1 operator, in the sense that the tools did not encounter the time-out issue and therefore they were able to provide proofs for all the corresponding statements.

Next, we moved to the conceptual part of the thesis. We presented moral luck and the problem that comes with it. In particular, it conflicts with the ethical principle that agents are not morally responsible for actions that are outside their control. The relation between moral luck and T-STIT I/O logic has been covered by the following question:

RQ4: It has been suggested that moral luck can be studied using deontic logic, but which aspects can be analyzed and which aspects are out of reach?

According to Nagel's studies, moral luck can be classified into four different kinds. Agents can be morally lucky due to their characters, circumstances or a combination of both, known as causal moral luck. However our logic is not able to capture aspects of an agent's character or circumstances. Thus, the class that we are able to focus on, is the one that deals with the agent's actions and their consequences, known as resultant moral luck. We identified which aspects of it can be captured by denotic logic. Looking at an example from resultant moral luck, we noticed that the agents always committed violations to the obligations. Finally examples of moral luck were formulated in terms of temporal STIT logic and we used I/O logic to determine what is obligatory in those scenarios. The operators out_1 and out_2 behaved as expected but the examples showed again the limits of unconstrained I/O logic. Because of violations to obligations, the output is inconsistent with the input set. To handle this issue, two possible approaches were presented in form of *constrained I/O logic* and an *I/O logic with consistency check*.

All the research questions could be answered to a satisfiable degree. However, those answers lead us to the identification of new problems for which we can formulate again new research questions for future work. The newly identified problems will be discussed in the final section of this thesis.

5.2 Future work

In terms of STIT logic, we considered Lorini's temporal STIT logic [22] because he provided a possible world semantics for it. The presented embedding is based on those and it could be realized by using techniques from previous work. But as presented, the implementation of the embedding also has its limitations in terms of being only able to deal with finite models. The original semantics of STIT logic are given in terms of branching time and agent choice structure and most of the examples from the literature are formulated in those. So far, there has been no work conducted with regard to those semantics in HOL and it is still open if it is possible to do so. Realizing an embedding based on the original semantics may overcome the limitations that we have faced. Further, we should also investigate the following: is it feasible to prove that Lorini's possible world semantics and the original ones are equivalent? For simple examples, it is straightforward to find equivalent models. But in some cases, it is not so evident and it is an open question if it is possible to find for all branching time and agent choice models a corresponding temporal Kripke STIT model. So a future step would then be to show the relationship between those two semantics and to prove that for every temporal Kripke STIT model, there is an equivalent model in terms of branching time and agent choice, and vice versa.

The implementation of the semantical embedding out_2 in HOL was explicitly tested on a variety of examples. We applied the operator to formulas of propositional logic and T-STIT logic. Even though we obtained a proof for every statement, this does not guarantee the correctness of the embedding. This also includes the embedding of the out_1 operator, presented in [7], which also has only been tested on examples so far. To do so, one has to provide the faithfulness of these embeddings. Therefore further work includes presenting proofs of the soundness and completeness of those. This also holds for the embedding of T-STIT logic in HOL.

This work presented an embedding of the out_2 operator but one may also be interested in providing embeddings for the other two traditional operators, out_3 and out_4. This would then complete the work of traditional or unconstrained I/O logic in HOL. Just like for the out_2 operator, there exists also a translation into modal logic for out_4 [23] and its formulation is similar to the one for out_2. In contrast to the formulation for out_2, for out_4 it is required that we consider a modal logic **S** such that **KT** $\subseteq$ **S** $\subseteq$ **KT45**. Thus the embedding for out_4 will be almost identical to the one for out_2, the only difference is that the relation R for the corresponding $\Box$ operator has to fulfill the property of reflexivity. As illustrated on an example in the previous chapter, the rule of cumulative transitivity does not hold for the operator out_2 and therefore $Nitpick$ generated a counter model. However when adding the condition of reflexivity to the relation for the lifted operator $\Box_l$, then $Nitpick$ is no longer able to disprove the corresponding statement and additionally $Sledgehammer$ provides us with a proof. Further, the examples for testing the rules of *AND*, *SI* and *WO* also hold for the out_4 operator. Still, one would need to test the operator on further examples and finally provide again the faithfulness of the embedding. Additionally, a lot of the statements could not be proven for the out_1 operator due to the complexity of the formulas, therefore one might consider to optimize the embedding of it or to use an alternative formulation for this operator.

After the completion of traditional I/O logic in HOL, the next step would be to move to the ”stronger” I/O logics. Due to the vulnerability of traditional I/O logic to violations which occur in the examples of moral luck, a next step would be to implement constrained I/O logic [24] into Isabelle/HOL by using again the semantical embedding approach. Because of the downside of having no proof theory, there have been suggestions in the I/O logic framework to overcome this. Some works [27] propose an unconstrained I/O logic with a consistency check, which allows the logic to handle violations, CTD, and dilemmas while still preserving its proof theory. So it would be interesting to see if embeddings for those kinds of I/O logics could be obtained as well.

To further analyze the principle of moral luck, T-STIT logic must be extended with additional operators. For the examples, the STIT operators were used to represent the actions and choices of the agents. To study the examples from moral luck more deeply, one may also have to consider the intentions of the agents or their knowledge. Knowledge operators exist in epistemic logic and there have also been attempts of adding these to STIT theory [13]. It may be interesting to see if it is possible to extend STIT logic with a notation to express an agent's intention. STIT logic is already able to express actions and abilities of agents, but together with an agency operator for intention, those operators can be used to formalize an agent's responsibility and blameworthiness which would be very useful when studying examples of moral luck.

Appendix A

.1 Deontic Logic

In deontic logic, $\bigcirc p$ is usually used in order to express that it is obligatory that p is the case, (or it ought to be that p is the case), and Pp to express that it is permitted, or permissible, that p is the case. The term 'deontic' originates from the ancient Greek word $de\grave{}on$, meaning that which is binding or proper.

The most familiar deontic logic is known as *Standard Deontic Logic (SDL)*, proposed by von Wright in 1951. It has been developed as a branch of modal logic where an obligation is interpreted as a variation of modal necessity. Instead of the unary operators for the alethic modalities necessity and possibility, denoted by $\Box$ and $\Diamond$, it uses unary operators for the deontological modalities of obligation and permission, denoted by $\bigcirc$ and P. Having this modal logical setting brings several advantages with it. In the first place, it allows us to make use of Kripke Semantics which means that obligations can be modeled with possible worlds techniques. Therefore, we say that a formula $\bigcirc p$ is true if and only if p is true in all *ideal* worlds. Further, it lets us define permissions in terms of obligations. To say that p is permissible can be stated as it is not obligatory that p is not the case. Formally, $Pp = \neg\bigcirc\neg p$. To deal with prohibition, an operator Fp, which is interpreted as it is forbidden that p is the case, is also defined with the help of the ought operator $\bigcirc$, namely $Fp = \bigcirc\neg p$. To say that p is forbidden is equivalent to say that it is obligatory that p is not the case. In terms of semantics, we have that a formula Pp is true if and only if there exists at least one *ideal* world in which p is true, and Fp is true if and only if p is false in all *ideal* worlds. Additionally, the axiomatization of this theory is pretty straightforward and the proof properties are well known. In particular, *SDL* is a propositional modal logic which extends the propositional tautologies with the axioms $K : \bigcirc(p \to q) \to (\bigcirc p \to \bigcirc q)$ and $D : \bigcirc p \to Pp$. Moreover it is closed under the primitive inference rules of *modus ponens:* if $\vdash p$ and $\vdash (p \to q)$ then $\vdash q$ and *Necessitation:* if $\vdash p$ then $\vdash \bigcirc p$ where the symbol $\vdash$ is read as 'is provable'. *Modus ponens* is also known as *detachment*. More specifically, in the literature of deontic logic, the term *factual detachment* is used. The resulting system is known as **KD**. The elegance of this theory lies in its simplicity. Being classified as a propositional modal logic, *SDL* is gifted with another benefit. Extension with other modalities such as temporal operations or an action modality can easily be realized. For instance, in his work [19], Horty investigated the combination of deontic logic with STIT logic, a modal logic of action.

But being a highly simplified theory has also its downside. A well-known problem that *SDL* has to deal with, is its vulnerability to deontic paradoxes. For instance, some paradoxes are related to weakening: $\bigcirc(p \wedge q) \rightarrow \bigcirc p$. As deontic logic have been developed as a branch of modal logic, most deontic logics support weakening. On the one hand, we have that $\bigcirc(p \wedge q) \rightarrow \bigcirc p$ might seem intuitive but on the other hand, its equivalent $\bigcirc p \rightarrow \bigcirc(p \vee q)$ does not look so convincing. An example of such paradoxes is Ross's paradox (1941). Consider the following two statements:

1. It is obligatory that the letter is mailed.
2. It is obligatory that the letter is mailed or the letter is burned.

In *SDL*, the statements 1 and 2 can be expressed by $\bigcirc m$ and $\bigcirc(m \vee b)$, respectively. Further, we have that $\vdash m \rightarrow (m \vee b)$. Applying the *Necessitation* rule to it, we get that $\vdash \bigcirc(m \rightarrow (m \vee b))$. Finally by the application of the K axiom and followed by the *modus ponens* rule, we can infer $\vdash \bigcirc m \rightarrow \bigcirc(m \vee b)$. This indicates that the second statement follows from the first one. However, it seems odd to say that an obligation to mail the letter entails an obligation that can be fulfilled by burning the letter. This is a counterintuitive derivation and it can be traced back to the interpretation of 'or' in our natural language.

Another main issue in the study of deontic logic is the representation of the so-called *Contrary-To-Duty* (CTD) obligations. Several well-known paradoxes are caused by those. A CTD obligation is an obligation which tells us what ought to be the case if something that should not be is the case. In *SDL*, a conditional obligation is normally formulated as $q \rightarrow \bigcirc p$, where q represents the condition and p the deontic conclusion. We say that the conditional obligation $q \rightarrow \bigcirc p$ is a *Contray-To-Duty* or *secondary* obligation of the *primary* obligation $\bigcirc p_1$, when q and p_1 are contradictory. Forrester's paradox, sometimes also referred to as the gentle murderer paradox, is a popular example. Consider the following scenario:

1. Smith should not kill Jones.
2. If Smith kills Jones then he should do it gently.
3. Smith kills Jones.

In *SDL*, those three statements are represented by the formulas $\bigcirc\neg k$, $k \rightarrow \bigcirc(k \wedge g)$ and k, respectively. They are considered as the set of premises. As $\neg k$ and k are contradictory, we have that the conditional obligation $k \rightarrow \bigcirc(k \wedge g)$ is a CTD obligation to the obligation $\bigcirc\neg k$. Since *SDL* supports the rule of *modus pones* or *factual detachment*, we can infer $\bigcirc(k \wedge g)$ from the statements 2 and 3. From $\bigcirc(k \wedge g)$, we are able to derive $\bigcirc k$ which leads us to the main problem of this paradox. We have that $\bigcirc k$ and $\bigcirc\neg k$ are inconsistent in *SDL* even though the three formulas from our set of premises were intuitively consistent. In *SDL*, there is no consistent formulation available for this paradox.

The type of possible worlds semantics used by *SDL* is not flexible enough. In this semantics, we are only able to distinguish between two types of worlds, namely *actual* and *ideal* worlds. From Forester's paradox, we had initially $\bigcirc\neg k$ and we are able to derive $\bigcirc k$ from a set of formulas. This causes then the paradox since it is clear that no ideal world can hold k and $\neg k$ at the same time. Thus ideal worlds are not enough in order to represent a proper model for this paradox. To do so, the notation of *sub-ideal* worlds is introduced and which solves then the inconsistency in the ideal worlds. Further, a preference ordering can be established on these sub-ideal worlds due to the finer distinction between a hierarchy of sub-ideal worlds instead of one type of ideal world. By replacing the principle of *ideality* of *SDL* by the principle of *optimality*, we move to *dyadic deontic logic (DDL)*, which has been introduced by Hansson. Hansson's proposed *DDL* is actually able to deal with contrary-to-duty reasoning.

A dyadic obligation is a conditional obligation, which is formulated as $\bigcirc(p \mid q)$ and is interpreted as it is obligatory that p is the case if q is the case. An unconditional obligation can also be formulated as follows $\bigcirc(p \mid \top)$. They are seen as special kind of dyadic obligations. $\{\bigcirc(k \mid \top), \bigcirc(k \wedge g \mid k), k\}$ represents a dyadic formulation of Forrester's paradox. Due to the optimally principle of *DDL*, a consistent formulation of the paradox can be achieved.

In contrast to *SDL* and *DDL*, which are analyzing the deontic modalities with reference to a set of possible worlds, there exist as well a family of frameworks which are referred to as *norm-based* deontic logic. A set of explicitly given norms is used in order to evaluate the deontic modalities. In such frameworks, the focus lies on interference patterns and thus the perspective is different from the traditional setting. In particular, for some given input A, referred to as *facts*, and a set of given conditional norms N, the framework tells us which norms apply. A instance of such a framework is Makinson's and van der Torre's I/O logic [23]. A key characteristic of this logic is its semantics. Rather than reusing truth-values and possible worlds techniques as other logics do, it provides *operational* semantics which are based on detachment. In order to capture the meaning of the deontic concepts, I/O operations are yielding outputs, seen as obligations, for inputs. This is achieved by defining different output operators, denoted by out. $x \in out(N, A)$ means than that 'given the input set A, x is obligatory under the norms N. In terms of proof theory, I/O logic consists of a set of inference rules. Those rules are applied to pairs of formulas rather than individual ones. I/O logic is covered in more detail in chapter 2.

Appendix B

.2 Ought-to-be vs Ought-to-do

When it comes to ought-to-be and ought-to-do sentences and obligations, philosophers often mention that there is clear distinction between both of them. 'It ought to be that you are friendly' denotes an ought-to-be sentence which expresses an ought-to-be obligation whereas 'You ought to help an injured person' represents an ought-to-do sentence which states an ought-to-do obligation. Deontic logic is primarily concerned with ought-to-be obligations, however in natural languages, besides expressing statements that something is ought to be, one also wants to have statements that someone ought to do something. The deontic operator $\bigcirc$ seems to capture the idea of the first concept, but to what extent can it be used to represents the second concept? In deontic logic, it often has been assumed that what an agent ought to do can be identified with the notation of what it ought to be what the agent does. If this assumption is correct then it is possible to paraphrase a sentence like 'You ought to go home' as 'It ought to be that you go home'. A way to formulate the concept of an ought-to-do obligation is to add some action logic or dynamic logic to a deontic system or vice versa. Then this would allow stating someone is ought to do something if and only if it ought to be that someone sees to it or brings it about.

Among deontic logicians and philosophers, the distinctions between the logics of ought-to-be and ought-to-do obligations is a highly discussed topic. In particular, we have that:

– Ought-to-do deontic statements refer to actions and express imperatives of the form "an agent ought to perform an action".

– Ought-to-be deontic statements express a desired state of affairs (results of actions) at a certain moment.

For Humberstone, there is also a distinction between the two kinds of ought statements. Humberstone [21] states that ought-to-be statements are *situational* oughts whereas ought-to-do statements are *agent-implicating* oughts. Horty illustrated Humberstone's distinction in his book [19]. He points out a scenario in which Albert has competed in a gymnastics event. From all the performances of the participants, Albert's one was clearly superior, however, the judge is known to be biased, so that it most likely that somebody else will be awarded the medal.

Then the sentence 'Albert ought to win the medal' is referred to that kind of statement that Humberstone would consider as a situational ought. It reflects a judgment about a situation, not about Albert, and can, therefore, be paraphrased as 'it ought to be that Albert wins the medal'. There is no indication, that it will be Albert's fault when he does not manage to win the medal. Further is winning the medal not within Albert's control after his performance. However, if one assumes that Albert has been lazy and ignoring his training schedule, then the sentence 'Albert ought to practice harder' would refer to the kind of ought statement that Humberstone categorizes as agent-implicating. It indicates that Albert has the ability to actually practice harder and the blame comes down to him if he fails to do so.

Further, Horty addresses the challenge on how to formulate obligations for actions in his book. In particular, the focus lies on the assumption whether that what an agent ought to do can be identified with the notion of what it ought to be what the agent does. As a first attempt, he defines an obligation to do an action as an obligation that such an action is done. In other words, he analyses whether ought-to-do can be reduced to ought-to-be. In order to formalize ought-to-do deontic statements, the STIT framework is extended with the Standard Deontic Logic (SDL) ought-operator, denoted by $\bigcirc$. Then he investigates the claim of the Meinong-Chisholm thesis which states the following:

> An agent α ought to see to it that φ, if and only if, it ought to be the case that the agent α sees to it that φ.

Finally, he comes to the conclusion that ought-to-do statements can't be formalized as ought-to-be statements about action. Thus, the statement "agent α ought to see to it that A" can't be captured by the formula $\bigcirc[\alpha\, cstit: A]$, which corresponds to the statement "it ought to be that agent α sees to it that A". The so-called gambling problem serves as an illustration in order to justify his claim.

.3 The gambling problem

We first give a quick overview on Horty's utilitarian STIT model before looking at the gambling problem. Such a model corresponds to the tree which represents a branching structure for indeterministic time. Each moment in the tree is represented as a partitioning of branches or histories. It is open to the future meaning that each branch represents a growth of the world. Horty uses a function $Value$ which assigns to each history a value in terms of a real number. A real number is representing the utility of that history. Hence the utility of a history h is denoted by $Value(h)$. The higher the value of the utility, the better the history. In Horty's

STIT model, propositions are evaluated for a pair composed of a moment and a history. A proposition A is true at a moment-history pair m/h if and only if it got assigned the value $True$ in the STIT model. Formally $m/h \in V(A)$ where V denotes the evaluation function mapping each proposition to a set of m/h pairs. The general evaluation for a formula $\bigcirc A$ in such a model is the following: $\bigcirc A$ holds at a moment-history pair m/h if and only if there is a history h' passing through that moment m such that A holds for all moment-history pairs m/h'' for which that utility of history h'' is at least as great as the utility of history h'.

The example of the gambling problem captures a moment m in which an agent faces the choice between gambling or refraining. By choosing to gamble, the agent might double his 5 dollars or might lose them. This is represented by the action or choice K_1. If the agents, however, decides to refrain, he keeps his 5 dollars and is represented by the action K_2. The statement A expresses that the agent gambles whereas, $\neg A$ indicates that the agent refrains. Figure 1 illustrates the situation.

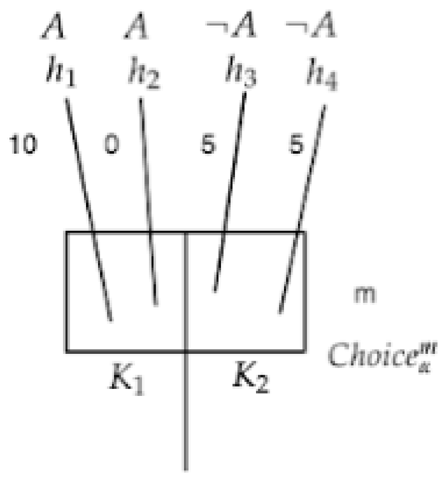


Figure 1: The gambling problem

The histories h_1 and h_2 represent the possible outcomes when the agent decides to gamble, i.e. to perform the action K_1. In one history, the agent doubles his money and ends up with 10 dollars whereas, in the other one, he ends up losing his money. By choosing to refrain and therefore not taking part in the game, the agent ends up either in the history h_3 or h_4, where the agent still preserves his 5 dollars.

Since $[\alpha\, cstit: A]$ is true at the index m/h_1 and also along all the histories with greater or equal utility, we have that the formula $\bigcirc[\alpha\, cstit: A]$ is settled true at moment m for this model. However an interpretation of $\bigcirc[\alpha\, cstit: A]$ as "agent α ought to perform action K_1" is counter-intuitive and a strange conclusion to draw here. By gambling, the agent may risk an outcome of utility 0 but at the same time, he is able to guarantee an outcome of utility 5 just by not taking part in the game.

From the description of the gambling problem, there is no information on which action is better than another one. Further, without knowing the probabilities of winning, there is nothing we can say in favor of the action K_1, except that this choice might be more preferred by the more adventurous agents. We just know that when performing K_1, we may either end up with more or less money then by choosing K_2.

This demonstrates that $\bigcirc[\alpha\, cstit: A]$ can't be interpreted as "agent α ought to see to it that A". So it is not sufficient to simply adapt and generalize the SDL ought-operator to STIT-logic in order to express ought-to-do statements.

Ought-to-be statements are evaluating the truth based on the optimal histories and this optimality is determined by the utilities associated with the individual histories. Having the highest utility is enough for a history to be considered as optimal. So if ought-to-be is concerned about optimal histories, then ought-to-do is dealing with optimal actions.

However actions are assumed to be non-deterministic, therefore actions correspond with sets of histories and one has to consider those rather than individual histories. The notation of optimality has to be adapted in such a way that it applies to sets of histories, which identify the different actions. The approach used by Horty is very simple and straightforward. The ordering of the action is based on the underlying ordering of the histories. More specifically, we say that an action is strictly better than another one if all the histories of that action are at least as good as the histories of the other one and it does not hold the other way around. Further an action K is categorized as *optimal*, if there is no other action K' such that K' is strictly better than K.

Having defined the ranking of actions, Horty introduces a utilitarian ought operator, also known as the dominance ought operator, which only applies to actions. This allows to formulate ought-to-do statements of the form "agent α ought to see to it that A" and is denoted by $\odot[\alpha\ cstit : A]$. $\odot[\alpha\ cstit : A]$ is settled true at moment m if and only if the outcomes of each optimal action, available to the agent α at moment m, guarantee the truth of A. When applying the new operator to the gambling problem, we have that both formulas $\odot[\alpha\ cstit : A]$ and $\odot[\alpha\ cstit : \neg A]$ are false, since neither actions are better or worse than the other one. More specifically, K_1 is not strictly better than K_2 since the utility of history h_2 is smaller than the utility of h_3, and K_2 is not strictly better than K_1 since the utility of history h_3 is smaller than the utility of h_1. This shows that none of the action is better than the other one and consequently both actions are categorized as optimal. Therefore we have that $\odot[\alpha\ cstit : A]$ is not valid since the outcomes of the action K_2 don't guarantee A, and $\odot[\alpha\ cstit : \neg A]$ does not hold since none of the outcomes of the action K_1 hold $\neg A$. Using this operator, in order to represent ought to do statements, solves the gambling problem. Still obtaining the utility 10 or 0 is not something that is control of the agent. None of the actions available to the agent can for sure obtain a utility of 10. The only factor which determines it is luck.

With the help of the following example, Horty points out that luck becomes an even more important factor when we consider multiple agents. In particular, it is about moral luck, the role of external factors in our moral evaluations. Assuming that we have two agents α and β, an action of agent β can be considered as an external factor for agent α since it is something that α is unable to control. Therefore achieving a certain utility depends not only on the actions of agent α but it also depends on what β does.

.4 Driving Example and Moral luck

The driving example [19] (Chapter 5, pages 119-121) addresses this challenge. In particular, Horty uses this example as an illustration to show the difference between the so-called dominance act utilitarianism and orthodox perspective on the agent's ought. The dominance act utilitarianism perspective is captured by $\odot[\alpha\ cstit : A]$. For the orthodox perspective, Horty introduces another ought-to-do operator, known as the orthodox ought operator $\oplus[\alpha\ cstit : A]$. In contrast to the dominance ought operator, the truth or falsity of $\oplus[\alpha\ cstit : A]$ may vary from index to index. Intuitively $\oplus[\alpha\ cstit : A]$ holds at an index if and only if the truth of A is guaranteed by each available action that is optimal under the circumstances in which the agent finds himself at that index.

Imagine a situation where we have two drivers who drive toward each other on a one-lane road. They have no possibility to stop or to communicate with each other, and at one particular moment, each one of them must independently decide whether he continues driving along the road or swerves. Further, the drivers can only swerve in one single direction, therefore they will end up in a collision when both of them choose to continue driving along the road or to swerve. A collision can only be avoided when one of them swerves and the other does not.

Figure 2 illustrates the driving example. We have agents α and β who represent the two drivers. Every driver has two actions available to him. K_1 represents the action that α swerves whereas K_2 refers to the action that α stays on the road. In a similar way, K_3 and K_4 denote the actions that β swerves or that β continues driving. Further, m represents the moment where both drivers have to make their

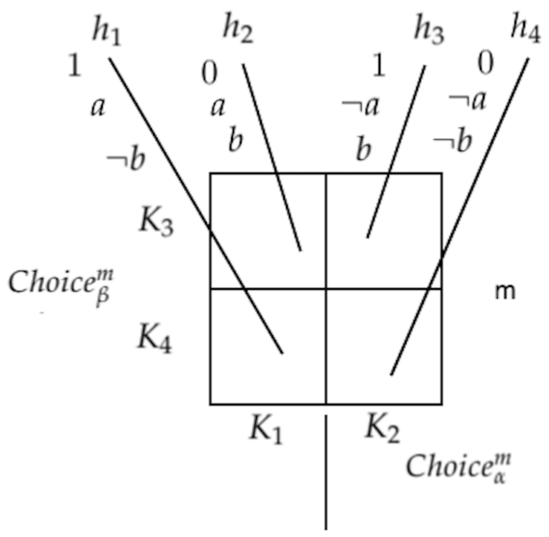


Figure 2: The driving example

choice. There are four histories, denoted by h_1, h_2, h_3 and h_4, passing through the moment m. The statement a, which expresses that α swerves, is true at the histories h_1 and h_2. Therefore h_1 and h_2 are the possible outcomes of the action K_1 whereas performing action K_2 might result either in the histories h_3 or h_4. Likewise the statement b, expressing that β swerves, holds at the histories h_2 and h_3, making them the possible outcomes of action K_3. Finally, β might end up in the histories h_1 or h_4 when he decides to perform K_4. The histories h_1 and h_3 are considered as the ideal outcomes since along those, one driver swerves and the other stays on the road and thus they don't end up in a collision. Therefore we assign the utility of 1 to those two histories. The other two histories h_2 and h_3 are non-ideal since along those the collision can't be avoided and thus they have a utility of 0.

To apply Horty's dominance ought operator, we first have to identify the optimal actions of the agents. For both agents, we have that both actions available are classified as optimal. Consequently, we have then for agent α, that $\odot[\alpha\ cstit : a]$ and $\odot[\alpha\ cstit : \neg a]$ are settled false at moment m since K_1 and K_2 are optimal but K_1 can only guarantee the truth of a whereas K_2 makes only $\neg a$ true. Likewise for β, we have that $\odot[\beta\ cstit : b]$ and $\odot[\beta\ cstit : \neg b]$ are also settled false at moment m. To evaluate the orthodox ought operator, we have to focus on an index which consists of a moment and an history. For instance, agent α finds himself at index m/h_1. At this particular index, agent β continues the road, therefore it reduces the possible outcomes of action K_1 to the history h_1 and of action K_2 to the history h_4. Therefore the optimal action available to agent α under these circumstances, is to swerve which is captured by action K_1. Since the truth of a is guaranteed by K_1, we have that the statement $\oplus[\alpha\ cstit : a]$ holds at the index m/h_1. However, when focusing on a different index, for instance $m/_2$, $\bigoplus[\alpha\ cstit : a]$ is no longer true. At this index, agent β swerves, thus the optimal action available to α is K_2, which refers to the action that α stays on the road. K_2 does not guarantee the truth of a but of $\neg a$, therefore at index m/h_2, the statement $\bigoplus[\alpha\ cstit : \neg a]$ holds.

According to Horty, the differences between those two ought operators provides us with another perspective on the issue of moral luck. Both operators are able to capture a legitimate sense of the ought used in our ordinary judgments. On the one hand, the orthodox ought captures the sense of looking back after the actual event. Imagine that at a later moment, through which history h_4 passes, agent α finds himself recovering from his injuries of the collision in the bed of a hospital. Since at the index m/h_4 it holds that $\bigoplus[\alpha\ cstit\ a]$, the agent α might regret his choice and might say to himself that he ought to have swerved. On the other hand, the dominance ought capture the sense of looking forward. Yet, the agent might honestly regret his choice, it is not something on which he can be blamed. From this perspective, neither was it the case that agent α ought to swerve nor was it the case that he ought to continue the road. This means that the agent did not fail to do something he ought to have done. Either choice of α at that time could have led to a collision and the outcome cannot be fully determined by himself. Also, the choice of the other agent plays a role in the determination of the outcome but this is a factor which is outside the control of agent α.

.5 Critical view on STIT theory

In computer science, actions are typically formalized in terms of preconditions and postconditions such that they can be put in a sequence plan. This allows for giving an operational semantics of a computer program like in Hoare logic. In a logic of action, propositions are not identified with actions, however, propositions are used to specify aspects of actions. On one hand, in dynamic logic actions are basically viewed as objects to which a name is assigned so that they are part of the logic's language. Postconditions indicate then the propositions that are made true by an action whereas preconditions tell us which propositions are necessary and sufficient conditions for the possible execution of an action. On the other hand, due to the different view of actions in STIT theory, it seems less clear on how to define the preconditions.

Horty's approach is based on the utilitarian perspective. Generally speaking, this means that an action is right or obligatory if it maximizes utility. The outcomes of the actions, or in other words, the histories, are expressed in terms of numbers. The only purpose of those numbers is to determine if a history is better or worse than another history. Other than that, those numbers have no meaning. For instance, for the gambling problem, the same results can be achieved when assigning different values to the histories but still preserving the same linear order. When changing the values of the histories h_1, h_2, h_3 and h_4 in the model to 1000, 7, 250 and 250, respectively, then the original model, depicted in 1, and the new one have no difference in terms of results. Both models are not distinguishable in the logic proposed by Horty.

Horty's approach to order the actions and how he identifies that a particular action is better than another one needs also to be discussed. Consider the following figure 3:

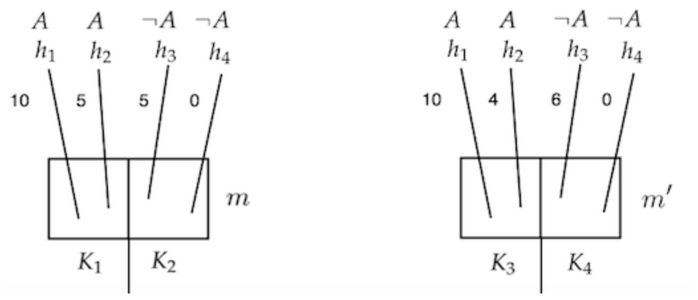


Figure 3: Two choice situations

On the left, we have a moment m where an agent α can either perform action K_1 or K_2. With Horty's ordering on actions, we get that K_1 is a better action than K_2 because all the histories of the action K_1 are at least as good as the histories of the action K_2 and it does not hold the other way around. Consequently, we have that $\odot[\alpha\, cstit:\ A]$ is true at m, so that α ought to perform K_1. However, for the situation on the right side, we have that at moment n, it is not that case that α ought to perform K_3 nor is it the case that he ought to perform K_4. In particular, K_3 is not a better action than K_4 since the history h_2, a possible outcome of K_3, is not better than h_3, a possible outcome of K_4. Similarly, K_4 is not a better action than K_3 due to the fact that for a possible outcome of K_4, for instance, h_3, there is a possible outcome of K_3 that is better, namely h_1. Because of the utilitarian setting, chosen by Horty, there are reasonable arguments to support the claim that K_3 is actually a better action than K_4. Like already mentioned before, the utilities, assigned to the individual histories, have no meaning. So all the information that we can get from the model on the right is the following: the highest utility can only be reached when performing the action K_1 whereas the lowest can be achieved by performing K_2. When an agent has to decide between those two actions, there are two valid reasons to choose K_1. In the first place, K_1 is the action that might lead to the best possible outcome, namely h_1. Second, by performing K_1, the agent can surely avoid the worst possible outcome, namely h_4. Horty's logic is only about choices, non-determinism, and utilities. The probabilities regarding the occurrence of the different histories are unknown and also not considered by the logic.

Appendix C

This section presents the syntax, semantics and axioms of Lorini's temporal STIT logic [22] and serves to provide background knowledge to the reader. All of the definitions from this section are taken from Lorini's work [22], otherwise we put the corresponding reference to that definition. Besides Lorini's possible world semantics, we will also give a semantics in *Branching Time and Agent Choice*, which is taken from Horty's work [19].

.6 Temporal STIT logic

Definition 1 (Syntax) Let $\mathbb{P}$ be the set of atomic propositions and Agt be a finite set of agents. The language $\mathcal{L}_{\text{T-STIT}}$ of temporal STIT logic is generated by the following BNF (Backus Normal Form):

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Box\varphi \mid [\alpha]\varphi \mid [Agt]\varphi \mid G\varphi \mid H\varphi$$

where $p \in \mathbb{P}$ and $\alpha \in Agt$.

Other connectives are introduced by the definitions:

disjunction	$\varphi \vee \psi$	is	$\neg(\neg\varphi \wedge \neg\psi)$
implication	$\varphi \to \psi$	is	$(\neg\varphi) \vee \psi$
equivalence	$\varphi \leftrightarrow \psi$	is	$(\varphi \to \psi) \wedge (\psi \to \varphi)$
versum	$\top$	is	$p \vee \neg p$ (for some $p \in \mathbb{P}$)
falsum	$\bot$	is	$\neg\top$

$\Box\varphi$ can be interpreted as "φ is true no matter what the agents do" or simply "φ is necessarily true". The dual of $\Box$ is defined as $\Diamond\varphi = \neg\Box\neg\varphi$ and expresses "φ is possibly true".

Operators of the form $[\alpha]$ represent the Chellas's STIT operators. The formula $[\alpha]\varphi$ captures the fact that φ is guaranteed by an action or choice of agent α and can be interpreted as "agent α sees to it that φ regardless what the other agents do". For simplification reasons, the interpretation of $[\alpha]\varphi$ can be shortened to "agent α sees to it that φ". The dual operator of $[\alpha]$ is defined in the usual way as: $\langle\alpha\rangle\varphi = \neg[\alpha]\neg\varphi$. The dual of the Chellas's STIT operator $\langle\alpha\rangle\varphi$ can be interpreted as "agent α allows it that φ". Further STIT theory offers the way to formalize the concept of ability. This is achieved by combining the $\Diamond$ operator with $[\alpha]$. The formula $\Diamond[\alpha]\varphi$ expresses then that agent α has the ability to see to it that φ or that agent α has the choice to enforce φ.

Remark Please note that in STIT literature, $[\alpha]\varphi$ is usually written $[\alpha\, cstit:\ \varphi]$, but the former notation is used because of conciseness reasons. Additionally, there exist several STIT operators such as the deliberative STIT operator which can be defined in terms of Chellas's STIT operator and the $\Box$ operator as following: $[\alpha\, dstit:\ \varphi] = [\alpha\, cstit:\ \varphi] \wedge \neg\Box\varphi$.

$[Agt]$ represents the group STIT operator. For a propositional formula φ, it captures the fact that φ is guaranteed by a choice of all the agents and has to be interpreted as "all agents see to it that φ by acting together". Just like for the Chellas's STIT operator, the dual operator of $[Agt]$ is expressed as $\langle Agt\rangle\varphi = \neg[Agt]\neg\varphi$.

Moreover, the operators G and H are tense operators that are used to express facts that are always true in the strict future respectively facts that are always true in the strict past. The strict future, respectively the strict past, does not include the present. $G\varphi$ can be interpreted as "φ will always be true in the future" whereas $H\varphi$ means "φ has always been true in the past". The dual operator of G is defined as $F\varphi = \neg G\neg\varphi$ and can be read as "φ will be true at some point in the future". Likewise, the dual operator of H is defined as $P\varphi = \neg H\neg\varphi$ and can be read as "φ has been true at some point in the past".

Finally, Lorini defines two very helpful operators as follows:

- $G^*\varphi = \varphi \wedge G\varphi$
- $F^*\varphi = \neg G^*\neg\varphi$

$G^*\varphi$ expresses that "φ is true in the present and will always be true" and $F^*\varphi$ means that "φ is true in the present and will be true at some point in the future".

.7 Semantics

This section presents two possible semantics for $\mathcal{L}_{\text{T-STIT}}$. The first alternative is a Branching Time (BT) structure augmented by an agent choice (AC) function, abbreviated BT+AC structure, and corresponds to the original framework for STIT logic given by Belnap [8]. However, some works, like the one proposed by Lorini [22], have also adapted a Kripke style semantics for STIT logic. Those have the advantage of being similar to the semantics of modal logic than Belnap's original semantics.

.7.1 Branching Time and Agent Choice

The semantics of STIT can be embedded in a branching time structure (BT), which is defined as:

Definition 2 (BT structure [19]) A branching time structure $\mathcal{M}$ is of the form $\langle M, < \rangle$, where:

- M is a non-empty set of moments
- $<$ is a transitive, irreflexive and treelike ordering on M.

By *treelike* ordering, we say that for any moments m_1, m_2, $m_3 \in M$, if $m_1 < m_3$ and $m_2 < m_3$ then either $m_1 = m_2$ or $m_1 < m_2$ or $m_2 < m_1$.

A *history* is considered as a maximal linearly ordered set of moments m from M. Further, when we have $m \in h$, we say that the moment m lies *on* the history h and the set of all the histories is denoted by $Hist$.

$$H_m = \{h \mid h \in Hist, m \in h\}$$

represent the set of histories passing through the moment m. A *moment-history pair* is a pair m/h consisting of a moment m from M and a history h from H_m. In other words, it's a history and a moment in that history.

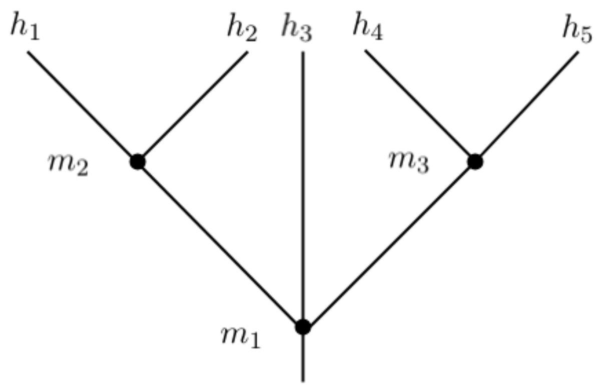


Figure 4: Branching Time

Figure 4 visualizes a branching time structure which is illustrated as a tree containing five histories $h_1 \dots h_5$. Further there are three moments, denoted by m_1, m_2 and m_3, lying on the different histories. For instance, we have that $m_2 \in h_1$ and $m_1 \in h_3$. The upward direction represents the forward direction of time and so we have that $m_1 < m_2$ and $m_1 < m_3$. Finally, we can also give the histories passing through a certain moment. For example, the set H_{m_3}, denoting the histories going through m_3, contains h_4 and h_5.

By augmenting the BT structure with the choices of agents, one obtains the most elementary framework in STIT logic. The resulting structure is called *agents and choices in branching time* and is abbreviated as BT+AC structure.

Definition 3 (BT + AC structure [19]) A branching time and agent choice (BT+AC) structure is a tuple $\mathcal{M} = \langle M, <, Choice, V \rangle$ where:

- $\langle M, < \rangle$ is a BT structure.
- $Choice : Agt \times M \mapsto 2^{2^{Hist}}$ is a function mapping each agent α and moment m into a partition $Choice^m_\alpha$ of the set of histories H_m.
- $V : \mathbb{P} \mapsto 2^{M \times Hist}$ is a valuation function assigning to each atom p a set $V(p) \subseteq M \times Hist$.

In general, a $BT + AC$ structure is often referred to simply as STIT structure or STIT model. The $Choice$ function is a very fundamental aspect of the BT+AC structure and it is specified detailed in Horty's book [19].

.7.2 Choice function

Individual Choice

Each equivalence class induced by $Choice^m_\alpha$ can be seen as a choice or action that is available to agent α at moment m. For any history $h \in H_m$, $Choice^m_\alpha(h)$ returns a particular choice K from $Choice^m_\alpha$ such that it contains the history h. Or in other words, the particular action that has been performed by agent i at the moment-history pair m/h. Formally, $Choice^m_\alpha(h) = \{h' \mid \exists K \in Choice^m_\alpha$ such that $h, h' \in K\}$. The histories belonging to a particular action K can be thought of as *possible outcomes* that might result from performing this action.

Further, there are several constraints imposed on the $Choice$ function. In order to impose the first constraint on the $Choice$ function, Horty introduces the following definition:

Definition 4 (Undivided histories [19]) Let $\langle M, < \rangle$ be a BT-structure such that $m \in M$ and $h_1, h_2 \in H_m$. h_1 and h_2 are said to be $undivided$ at m if and only if there exists $m' \in M$ such that $m < m'$ and $m' \in h_1 \cap h_2$.

Then he defines the property of *No Choice between undivided histories*, which forms then the first constraint on the $Choice$ function.

Definition 5 (No choice between undivided histories [19]) $\forall h, h' \in H_m$, if h and h' are undivided at the moment m, then $h' \in Choice^m_i(h)$ for every agent i.

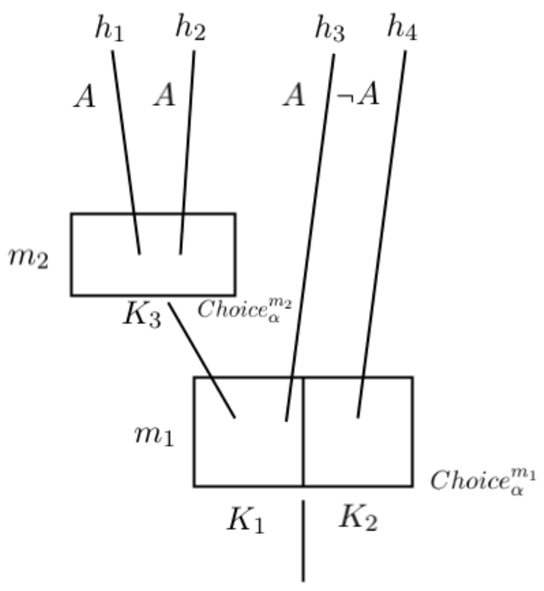


Figure 5: Branching Time with an agent's choice

Figure 5 illustrates the choice partitions. At moment m_1, there are two choices available to agent α, namely K_1 and K_2. Formally, we write $Choice_\alpha^{m_1} = \{K_1, K_2\}$ with $K_1 = \{h_1, h_2, h_3\}$ and $K_2 = \{h_4\}$. We can say, for instance, at the index m_1/h_3, the agent α performs action K_1 which might result in history h_1, h_2 or h_3. Further, the particular choice that contains the history h_3 is K_1, so we write $Choice_\alpha^{m_1}(h_3) = \{h_1, h_2, h_3\} = K_1$. Moreover, the histories h_1 and h_2 are undivided at moment m_1 since they share the later moment m_2 in common. Thus, we have $h_1, h_2 \in H_{m_1}$ and there is m_2 such that $m_1 < m_2$ and $m_2 \in h_1 \cap h_2$.

Group Choice

When dealing with group agency, Horty further specifies the $Choice$ function. In [19], he defines the so called *group actions* by introducing an *action selection* function s_m from Agt into 2^{H_m} such that for each $m \in M$ and $\alpha \in Agt$, $s_m(\alpha) \in Choice_\alpha^m$. So, the selection function s_m selects a particular action for each agent at moment m. Next, the set $Select_m$ is the set containing all of the action selection functions s_m.

$$Select_m = \{s_m : Agt \mapsto 2^{H_m} \mid s_m(\alpha) \in Choice_\alpha^m, \forall \alpha \in Agt\}$$

This allows then to formulate the property of *independence of agents* (or *independence of choices*) and will be used as the second constraint. It can be interpreted in the sense that agents can never be deprived of choices due to the choices made by the other agents. Alternatively, this constraints means that every possible choice of an agent is consistent with every choice made by every other agent. Two choices are said to be consistent if and only if they have at least one outcome in common. This condition can be defined as:

Definition 6 (Independence of agents [19]) For each function $s_m \in Select_m$, we must have $\bigcap_{\alpha \in Agt} s_m(\alpha) \neq \emptyset$.

By using the action selection functions s_m, the $Choice$ function can be generalized in order to apply for particular groups of agents. The collective choice for a group of agents $A \subseteq Agt$ at a moment m can be stated as:

$$Choice^m_A = \{\bigcap_{\alpha \in A} s_m(\alpha) \mid s_m \in Select_m\}$$

Again, $Choice^m_A(h) = \{h' \mid \exists K \in Choice^m_A \text{ such that } h, h' \in K\}$.

To illustrate the concept of group choices, consider the example represented in figure 6 which can be found in Horty's book [19] (Chapter 2, page 30).

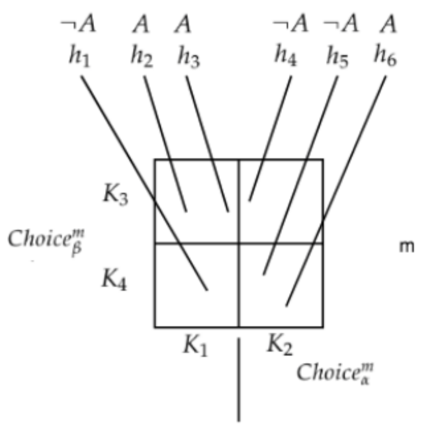


Figure 6: Branching Time with group choices

In contrast to the previous figure, we have two agents, α and β. Each one of them has several choices open to him/her. The choices available to agent α at the moment m are represented by the vertical partitions on H_m whereas the choice available to agent β at moment m are represented by the horizontal partitions on H_m. In particular, $Choice^m_\alpha = \{K_1, K_2\}$ with $K_1 = \{h_1, h_2, h_3\}$ and $K_2 = \{h_4, h_5, h_6\}$ and $Choice^m_\beta = \{K_3, K_4\}$ with $K_3 = \{h_2, h_3, h_4\}$ and $K_4 = \{h_1, h_5, h_6\}$. At moment m, each agent has two possible choices, so the set $Select_m$ will contain the following four functions s_1, s_2, s_3 and s_4:

$$\begin{array}{lcl} s_1(\alpha) = K_1 & \text{and} & s_1(\beta) = K_3 \\ s_2(\alpha) = K_1 & \text{and} & s_2(\beta) = K_4 \\ s_3(\alpha) = K_2 & \text{and} & s_3(\beta) = K_3 \\ s_4(\alpha) = K_2 & \text{and} & s_4(\beta) = K_4 \end{array}$$

Now let $A = \{\alpha, \beta\}$ denoting the group consisting of agent α and agent β. The choices available to the group A at moment m are then:

$$\begin{aligned} Choice_A^m &= \{\bigcap_{i \in A} s_m(i) \mid s_m \in Select_m\} \\ &= \{s_m(\alpha) \cap s_m(\beta) \mid s_m \in Select_m\} \\ &= \{s_1(\alpha) \cap s_1(\beta), s_2(\alpha) \cap s_2(\beta), s_3(\alpha) \cap s_3(\beta), s_4(\alpha) \cap s_4(\beta)\} \\ &= \{K_1 \cap K_3, K_1 \cap K_4, K_2 \cap K_3, K_2 \cap K_4\} \end{aligned}$$

For example, take m/h_3 as an index, then the group A is able to perform $K_1 \cap K_3$ which might result in the history h_2 or h_3. And since $K_1 \cap K_3$ is the particular action performed by A at m that contains the history h_3, we have $Choice_A^m(h_3) = K_1 \cap K_3$.

.7.3 Satisfaction in BT+AC structure

Definition 7 (Satisfaction [19]) Given a BT+AC structure $\mathcal{M} = \langle M, <, Choice, V \rangle$ and a moment-history pair m/h, we define the satisfaction relation $\mathcal{M}, m/h \vDash \varphi$ (read as "φ is true at the moment-history pair m/h in $\mathcal{M}$") as follows:

- $\mathcal{M}, m/h \vDash p \Longleftrightarrow m/h \in V(p)$
- $\mathcal{M}, m/h \vDash \neg\varphi \Longleftrightarrow \mathcal{M}, m/h \nvDash \varphi$
- $\mathcal{M}, m/h \vDash \varphi \wedge \psi \Longleftrightarrow \mathcal{M}, m/h \vDash \varphi$ and $\mathcal{M}, m/h \vDash \psi$
- $\mathcal{M}, m/h \vDash \Box\varphi \Longleftrightarrow \forall h' \in H_m : \mathcal{M}, m/h' \vDash \varphi$
- $\mathcal{M}, m/h \vDash \Diamond\varphi \Longleftrightarrow \exists h' \in H_m : \mathcal{M}, m/h' \vDash \varphi$
- $\mathcal{M}, m/h \vDash [\alpha]\varphi \Longleftrightarrow \forall h' \in Choice_\alpha^m(h) : \mathcal{M}, m/h' \vDash \varphi$
- $\mathcal{M}, m/h \vDash [Agt]\varphi \Longleftrightarrow \forall h' \in Choice_{Agt}^m(h) : \mathcal{M}, m/h' \vDash \varphi$
- $\mathcal{M}, m/h \vDash G\varphi \Longleftrightarrow \forall m' \in h$ such that $m < m' : \mathcal{M}, m'/h \vDash \varphi$
- $\mathcal{M}, m/h \vDash H\varphi \Longleftrightarrow \forall m' \in h$ such that $m' < m : \mathcal{M}, m'/h \vDash \varphi$

A formula φ is *valid* if and only if for every BT+AC models $\mathcal{M}$ and for every moment-history pairs m/h, we have $\mathcal{M}, m/h \vDash \varphi$. A formula φ is *satisfiable* if and only if there is a BT+AC model $\mathcal{M}$ and some moment-history pairs m/h such that $\mathcal{M}, m/h \vDash \varphi$.

Example 8 In Figure 5, we have that $\mathcal{M}, m_1/h_1 \vDash [\alpha]A$ since $Choice^{m_1}_{\alpha}(h_1) = \{K_1\} = \{h_1, h_2, h_3\}$ and for every history $h \in Choice^{m_1}_{\alpha}(h_1)$, we have $\mathcal{M}, m_1/h \vDash A$. However at the index m_1/h_4, the formula $[\alpha]A$ is not satisfied (i.e. $\mathcal{M}, m_1/h_4 \nvDash [\alpha]A$) since $Choice^{m_1}_{\alpha}(h_4) = \{K4\} = \{h_4\}$ and $\mathcal{M}, m1/h_4 \nvDash A$. Moreover, we have that $\mathcal{M}, m_2/h_1 \vDash \Box A$, because for every history $h \in H_{m_2} = \{h_1, h_2\}$, we have $\mathcal{M}, m_2/h \vDash A$. Finally, we have that $\mathcal{M}, m_1/h_1 \vDash GA$ since $m_2 \in h_1$ such that $m_1 < m_2$ and $\mathcal{M}, m_2/h_1 \vDash A$.

Example 9 In Figure 6, it is not possible for both agents to individually see to it that A, since $\forall h \in H_m$, we have $\mathcal{M}, m/h \nvDash [\alpha]A$ resp. $\mathcal{M}, m/h \nvDash [\beta]A$. But by acting together, the formula $[\mathcal{A}]A$ is true at index m/h_2 and m/h_3, where $\mathcal{A}$ denotes the group of agent consisting of α and β, is valid. For every history $h \in Choice^{m}_{\mathcal{A}}(h_3) = \{h_2, h_3\}$, we have $\mathcal{M}, m/h \vDash A$. Finally, we have that $\forall h \in H_m$, $\mathcal{M}, m/h \vDash \Diamond[\mathcal{A}]A$.

.7.4 Temporal Kripke STIT Model

In this section, we discuss a Kripke-style semantic for STIT logic. In contrast to the BT+AC semantics, the Kripke semantic for STIT takes the concept of worlds as a primitive. Let's first remind the definition of a Kripke model for modal logic.

Definition 8 (Kripke Model) A Kripke model is a structure $\mathcal{M} = \langle W, R, V \rangle$ where

- W is a nonempty set of possible worlds.
- R is a binary relation on worlds, called accessibility relation. For any world $w \in W$, let $R(w) = \{v \in W \mid (w, v) \in R\}$.
- $V : \mathbb{P} \mapsto 2^W$ is a valuation function. A proposition p is true at a world w if and only if $w \in V(p)$.

A Kripke model is illustrated in Figure 7. The possible worlds are represented by the nodes and the relation is indicated by the arrows from nodes to nodes. Here we have $W = \{w_1, w_2, w_3, w_4\}$ and for instance $R(w_1) = \{w_2, w_3\}$. Additionally, each possible world w is labelled with the propositional atom which are true at

w. The set of atoms $\mathbb{P}$ is composed of p and q, which gives us in this case the following valuation functions: $V(p) = \{w_1, w_2, w_3\}$ and $V(q) = \{w_1, w_4\}$.

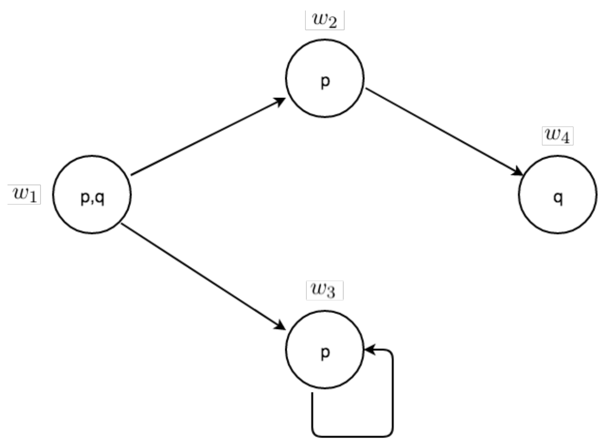


Figure 7: Standard Kripke Model

The temporal Kripke STIT model can be seen as Kripke model with multiple relations on which several constraints are imposed. A world w corresponds to a moment-history pair m/h in a BT+AC model. Moments will be defined as equivalence classes which are induced by an equivalence relation over the set of worlds and an agent's set of choice at a moment is a partition of that moment.

Definition 9 (Equivalence relation) Let R be a binary relation on the set W. R is called an $equivalence$ relation if it has the following three properties:

- Reflexivity: $\forall w \in W$, we have $(w, w) \in R$.
- Symmetry: $\forall w, v \in W$, if $(w, v) \in R$, then $(v, w) \in R$.
- Transitivity: $\forall w, v, u \in W$, if $(w, v) \in R$ and $(v, u) \in R$, then $(w, u) \in R$.

Further, given two binary relation R_1 and R_2, let $R_1 \circ R_2$ denote the operation of composition between binary relation. Finally, we define R^{-1}, the inverse relation of R, as $R^{-1} = \{(v, w) \mid (w, v) \in R\}$.

Before giving the definition of a temporal Kripke STIT model, we will list the different accessibility relations and give an interpretation for each of them.

For any given world $w \in W$, we have

- $R_\Box(w)$ defines the set of worlds that are alternatives to the world w. The equivalence classes induced by $R_\Box$ can be seen as moments. For any world $v \in W$, $v \in R_\Box(w)$ can be interpreted as "v and w belong to the same moment".

- $R_i(w)$ defines the set of worlds that identifies agent i's actual choice or action at world w. In other words, the set of all alternatives that are forced by agent i's choice or action at world w.
- $R_{Agt}(w)$ defines the set of worlds that identifies the actual choice or action of a group Agt at the world w. In other words, the set of all alternatives that are forced by the collective choice or action of all the agents at world w.
- $R_G(w)$ defines the set of worlds that are in the strict future of the world w. The strict future does not include the present.
- $R_H(w)$ defines the set of worlds that are in the strict past of the world w. The strict past does not include the present.

Lorini defines then the temporal Kripke STIT model as follows.

Definition 10 (Temporal Kripke STIT model) A temporal Kripke STIT model $\mathcal{M}$ is a tuple $\langle W, R_\Box, \{R_i \mid i \in Agt\}, R_{Agt}, R_G, R_H, V\rangle$ where:

- W is a non-empty set of possible worlds.
- $R_\Box$, every R_i and R_{Agt} are equivalence relations between the worlds in W such that:

  (C1) $R_i \subseteq R_\Box$

  (C2) $\forall w_1, \ldots, w_n \in W$: if $(w_i, w_j) \in R_\Box\ \forall i, j \in \{1, \ldots, n\}$ then $\bigcap_{i \in Agt} R_i(w_i) \neq \emptyset$

  (C3) $\forall w \in W : R_{Agt}(w) = \bigcap_{i \in Agt} R_i(w)$

- R_G and R_H are relations between the worlds in W such that R_G is serial and transitive, R_H is the inverse relation of R_G.

  (C4) $\forall u, v, w \in W$: if $u, v \in R_G(w)$ then $u \in R_G(v)$ or $v \in R_G(u)$ or $u = v$

  (C5) $\forall u, v, w \in W$: if $u, v \in R_H(w)$ then $u \in R_H(v)$ or $v \in R_H(u)$ or $u = v$

  (C6) $R_G \circ R_\Box \subseteq R_{Agt} \circ R_G$

  (C7) $\forall w \in W$: if $v \in R_\Box(w)$ then $v \notin R_G(w)$

- $V : \mathbb{P} \mapsto 2^W$ is a valuation function assigning to each atom p a set $V(p) \subseteq W$.

Constraint $(C1)$ states that an agent can only choose between possible alternatives. For any world $w \in W$, the equivalence relation R_i induces a partition on the set $R_\Box(w)$ and any element of this partition can be seen as a possible choice for an agent i at world w.

Constraint $(C2)$ corresponds to the property of *independence of agents* or *independence of choices*.

Constraint $(C3)$ means that the choices of agents in the group Agt are made up of the choices of each individual agent and no more.

Constraint $(C4)$ guarantees that the time is connected towards the future while constraint $(C5)$ ensures the connection towards the past.

Constraint $(C6)$ corresponds to the property of *no choice between undivided histories*. If v is in the future of w and u and v are in the same, then there exists an alternative z in the collective choice of all agents at w such that u is in the future of z.

Constraint $(C7)$ makes sure that if two worlds belong to the same moment then it is not possible that the one world is in the future of the other one.

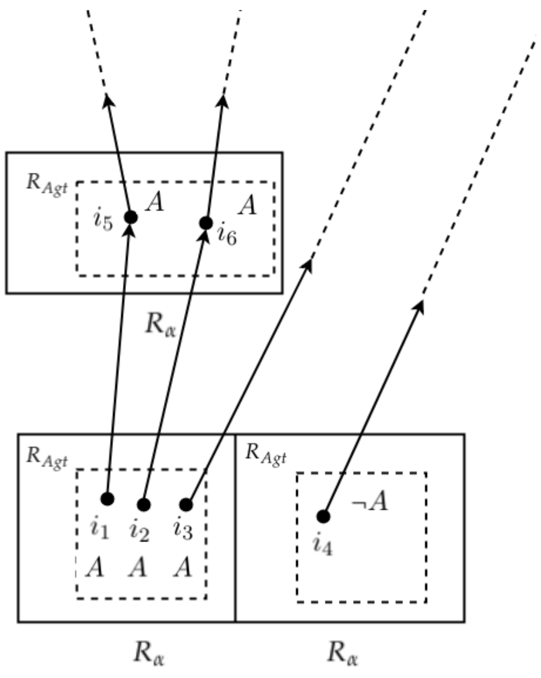


Figure 8: Temporal Kripke STIT Model

Figure 8 gives an illustration of a temporal Kripke STIT model. In fact, it is the same as shown in Figure 5 but just represented in Kripke STIT semantics. The moments, which are induced by the relation $R_\Box$, are represented by the rectangles. We have two moments, m_1 consists of the set of worlds $\{i_1, i_2, i_3, i_4\}$ whereas m_2 of $\{i_5, i_6\}$.The choices available to agent α at a moment are represented by columns. For example, at moment m_1, agent α has two choices, namely $\{i_1, i_2, i_3\}$ and $\{i_4\}$. The dotted rectangles show the choices available to the group. In this case, the group of agents only consists of agent α. The arrows serve as a representation for the temporal relation R_G.

Definition 11 (Satisfaction) Given a temporal Kripke STIT model $\mathcal{M} = \langle W, R_\Box, \{R_i \mid i \in Agt\}, R_{Agt}, R_G, R_H, V\rangle$ and a world w ∈ W, Lorini defines the satisfaction relation $\mathcal{M}, w \models \varphi$ (read as "φ is true at world w in $\mathcal{M}$") as follows:

- $\mathcal{M}, w \models p \iff w \in V(p)$
- $\mathcal{M}, w \models \neg\varphi \iff \mathcal{M}, w \not\models \varphi$
- $\mathcal{M}, w \models \varphi \wedge \psi \iff \mathcal{M}, w \models \varphi$ and $\mathcal{M}, w \models \psi$
- $\mathcal{M}, w \models \Box\varphi \iff \forall v \in R_\Box(w) : \mathcal{M}, v \models \varphi$
- $\mathcal{M}, w \models \Diamond\varphi \iff \exists v \in R_\Box(w) : \mathcal{M}, v \models \varphi$
- $\mathcal{M}, w \models [i]\varphi \iff \forall v \in R_i(w) : \mathcal{M}, v \models \varphi$
- $\mathcal{M}, w \models [Agt]\varphi \iff \forall v \in R_{Agt}(w) : \mathcal{M}, v \models \varphi$
- $\mathcal{M}, w \models G\varphi \iff \forall v \in R_G(w) : \mathcal{M}, v \models \varphi$
- $\mathcal{M}, w \models H\varphi \iff \forall v \in R_H(w) : \mathcal{M}, v \models \varphi$

A formula φ is $valid$ if and only if for every temporal Kripke STIT model $\mathcal{M}$ and for every world $w \in W$, we have $\mathcal{M}, w \models \varphi$. A formula φ is $satisfiable$ if and only if there is a temporal Kripke STIT model $\mathcal{M}$ and some world $w \in W$ such that $\mathcal{M}, w \models \varphi$.

Example 10 In Figure 8, we have for instance that $\mathcal{M}, w_1 \models [\alpha]A$ since $R_\alpha(w_1) = \{w_1, w_2, w_3\}$ and for every world $w \in R_\alpha(w_1)$, we have $\mathcal{M}, w \models A$. Further $\mathcal{M}, w_5 \models \Box A$, since for every world $w \in R_\Box(w_5) = \{w_5, w_6\}$, we have $\mathcal{M}, w \models A$. Finally, we have that $\mathcal{M}, w_1 \models GA$ since for every world $w \in R_G(w_1) = \{w_5\}$, we have $\mathcal{M}, w \models A$.

.8 Axioms

Finally, Lorini proposed the following axioms for T-STIT logic. First, we have all the tautologies of classical proposition calculus. For instance, a formula such as $\varphi \vee \neg\varphi$ is considered as a tautology.

The relation $R_\Box$, every R_i and R_{Agt} are defined as equivalence relation for the operators $\Box$, $[i]$ for each $i \in Agt$, and $[Agt]$, respectively. Thus we have all principles of a modal logic **S5** for those operators. In particular, for an operator $\triangledown \in \{\Box, [i]$ for each $i \in Agt, [Agt]\}$, the following principles are valid:

(Nec) if $\vdash \varphi$ then $\vdash \triangledown\varphi$

(K) $\vdash \triangledown(\varphi \to \psi) \to (\triangledown\varphi \to \triangledown\psi)$

(T) $\vdash \triangledown\varphi \to \varphi$

(B) $\vdash \varphi \to \triangledown\neg\triangledown\neg\varphi$

(4) $\vdash \varphi \to \triangledown\triangledown\varphi$

Remark We write $\vdash \varphi$ in order to express the formula φ is derivable. Alternatively, one also says that φ is a theorem or that φ is provable.

The temporal operator G satisfies all the principles of a modal logic **KD4** since the corresponding relation R_G is defined as serial and transitive. More precisely, besides the axioms (Nec), (K) and (4), we also have the following principle for the operator G:

(D) $\vdash \neg(G\varphi \wedge G\neg\varphi)$

The relation R_H is declared as the inverse relation of R_G, therefore the other temporal operator H just holds the principles of a modal logic **K**, which includes the axioms (Nec) and (K).

Further, Lorini gave the following axioms with respect to the class of temporal Kripke STIT models:

$(\Box \to i)$ $\vdash \Box\varphi \to [i]\varphi$

$(i \to Agt)$ $\vdash ([1]\varphi_1 \wedge \cdots \wedge [n]\varphi_n) \to [Agt](\varphi_1 \wedge \cdots \wedge \varphi_n)$

(AIA) $\vdash (\Diamond[1]\varphi_1 \wedge \cdots \wedge \Diamond[n]\varphi_n) \to \Diamond([1]\varphi_1 \wedge \cdots \wedge [n]\varphi_n)$

$(Conv_{G,H})$ $\vdash \varphi \to GP\varphi$

$(Conv_{H,G})$ $\vdash \varphi \to HF\varphi$

$(Connected_G)$ $\vdash PF\varphi \to (P\varphi \vee \varphi \vee F\varphi)$

$(Connected_H)$ $\vdash FP\varphi \to (P\varphi \vee \varphi \vee F\varphi)$

$(NCUH)$ $\vdash F\Diamond\varphi \to \langle Agt\rangle F\varphi$

(MP) if $\vdash \varphi$ and $\vdash \varphi \to \psi$ then $\vdash \psi$

(IRR) if $\vdash (\Box\neg p \wedge \Box(Gp \wedge Hp)) \to \varphi$ then $\vdash \varphi$, provided p does not occur in φ

For instance, the axiom $(\Box \to i)$ says that if φ is settled as true no matter what the agent does, then the agent sees to it that φ. Together with (AIA), they form two essential principles of Xu's axiomatization [31] for Chellas's STIT operator. According to Axiom $(i \to Agt)$, all agents bring about together what each of them brings about individually.

$(Connected_G)$ and $(Connected_H)$ are the axioms that guarantee the linearity of the future and the past, respectively. According to $(Conv_{G,H})$ and $(Conv_{H,G})$, we have that what is, will always have been and what is, has always been going to be, respectively. Both correspond to the basic axioms of interaction between future and time in minimal tense logic.

Axiom $(NCUH)$ states that if at some moment in the future it is possible that φ, then the collective group of agents will possibly reach a state in which φ is true. This axiom corresponds to the constraint of 'no choice between undivided histories'.

Moreover, we have the modus pones rule (MP), which says that if φ is derivable as well as $\varphi \to \psi$ is derivable, then ψ is derivable.

(IRR) corresponds to an alternative formulation of Gabbay's well-known irreflexivity rule. It has commonly been used in order to prove the completeness of several temporal logic where the time is assumed to be irreflexive. In temporal STIT logic, we also have a special kind of irreflexivity for the relation R_G, which is a consequence from the fact that $R_\Box$ is reflexive and the constraint $(C7)$ from definition 10.

Appendix D

.9 Examples

In his book [19], Horty illustrates group agency and ability through an example (Chapter 2, p. 30, Figure 2.5). With our embedding, it is possible to formulate the model of this example in Isabelle/HOL and verify the possible actions of the individual agents as well as the group actions.

The example is given in a BT+AC model but since the embedding is based on Kripke semantics, the example has to be transformed. In this case, it is pretty straightforward. The figure 9 represents the example in temporal Kripke STIT semantics.

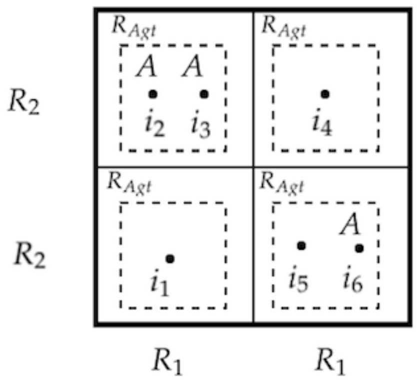
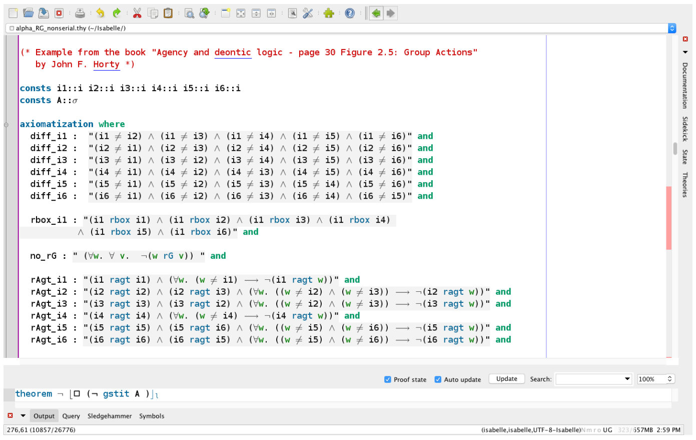


Figure 9: Group Actions

The figure 10 shows the first part of the implementation for the example in Isabelle/HOL. We start by defining our constants, in this case, the six possible worlds i_1, i_2, i_3, i_4, i_5 and i_6 of type i and the proposition A of type σ. It is important that we explicitly state that all the six worlds are different from each other. Further all the worlds in our model belong to the same moment, which can be stated for instance as $R_\Box(i_1) = \{i_1, i_2, i_3, i_4, i_5, i_6\}$, and since there is only a single moment, we have no two worlds w and v such that $v \in R_G(w)$. Next, we specify the relation R_{Agt} for every world. For instance at world i_2, we have $R_{Agt}(i_2) = \{i_2, i_3\}$.



alpha_RG_nonserial.thy (~/Isabelle/)

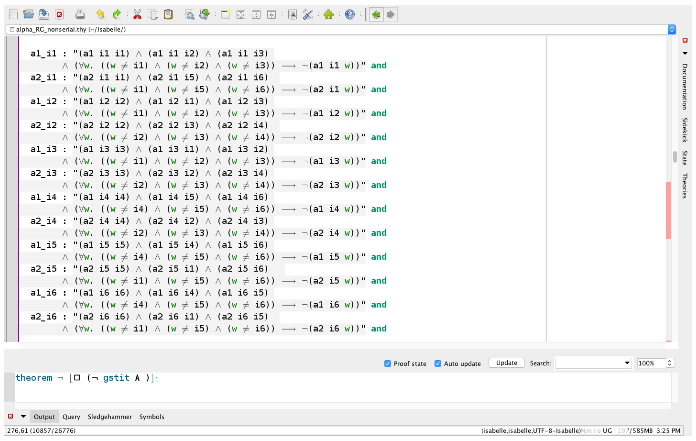
```
(* Example from the book "Agency and deontic logic - page 30 Figure 2.5: Group Actions"
   by John F. Horty *)

consts i1::i i2::i i3::i i4::i i5::i i6::i
consts A::σ

axiomatization where
  diff_i1 :  "(i1 ≠ i2) ∧ (i1 ≠ i3) ∧ (i1 ≠ i4) ∧ (i1 ≠ i5) ∧ (i1 ≠ i6)" and
  diff_i2 :  "(i2 ≠ i1) ∧ (i2 ≠ i3) ∧ (i2 ≠ i4) ∧ (i2 ≠ i5) ∧ (i2 ≠ i6)" and
  diff_i3 :  "(i3 ≠ i1) ∧ (i3 ≠ i2) ∧ (i3 ≠ i4) ∧ (i3 ≠ i5) ∧ (i3 ≠ i6)" and
  diff_i4 :  "(i4 ≠ i1) ∧ (i4 ≠ i2) ∧ (i4 ≠ i3) ∧ (i4 ≠ i5) ∧ (i4 ≠ i6)" and
  diff_i5 :  "(i5 ≠ i1) ∧ (i5 ≠ i2) ∧ (i5 ≠ i3) ∧ (i5 ≠ i4) ∧ (i5 ≠ i6)" and
  diff_i6 :  "(i6 ≠ i1) ∧ (i6 ≠ i2) ∧ (i6 ≠ i3) ∧ (i6 ≠ i4) ∧ (i6 ≠ i5)" and

  rbox_i1 : "(i1 rbox i1) ∧ (i1 rbox i2) ∧ (i1 rbox i3) ∧ (i1 rbox i4)
            ∧ (i1 rbox i5) ∧ (i1 rbox i6)" and

  no_rG : " (∀w. ∀ v.  ¬(w rG v)) " and

  rAgt_i1 : "(i1 ragt i1) ∧ (∀w. (w ≠ i1) ⟶ ¬(i1 ragt w))" and
  rAgt_i2 : "(i2 ragt i2) ∧ (i2 ragt i3) ∧ (∀w. ((w ≠ i2) ∧ (w ≠ i3)) ⟶ ¬(i2 ragt w))" and
  rAgt_i3 : "(i3 ragt i3) ∧ (i3 ragt i2) ∧ (∀w. ((w ≠ i2) ∧ (w ≠ i3)) ⟶ ¬(i3 ragt w))" and
  rAgt_i4 : "(i4 ragt i4) ∧ (∀w. (w ≠ i4) ⟶ ¬(i4 ragt w))" and
  rAgt_i5 : "(i5 ragt i5) ∧ (i5 ragt i6) ∧ (∀w. ((w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(i5 ragt w))" and
  rAgt_i6 : "(i6 ragt i6) ∧ (i6 ragt i5) ∧ (∀w. ((w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(i6 ragt w))" and
```

```
theorem ¬ ⌊□ (¬ gstit A )⌋ᵢ
```

276,61 (10857/26776) (isabelle,isabelle,UTF-8-Isabelle)

Figure 10: Isabelle/HOL embedding for the Group Action Example (part 1)

Moreover, we have to define the individual choices at the different worlds for the agents. In the example, we have two agents, a_1 and a_2. Taking for instance world i_2 again and let R_1 resp. R_2 denote the accessibility relations for the agent a_1 resp. a_2, then we have $R_1(i_2) = \{i_1, i_2, i_3\}$ and $R_2(i_2) = \{i_2, i_3, i_4\}$. The Isabelle/HOL formulation for those relations is shown in Figure 11.



alpha_RG_nonserial.thy (~/Isabelle/)

```
  a1_i1 : "(a1 i1 i1) ∧ (a1 i1 i2) ∧ (a1 i1 i3)
          ∧ (∀w. ((w ≠ i1) ∧ (w ≠ i2) ∧ (w ≠ i3)) ⟶ ¬(a1 i1 w))" and
  a2_i1 : "(a2 i1 i1) ∧ (a2 i1 i5) ∧ (a2 i1 i6)
          ∧ (∀w. ((w ≠ i1) ∧ (w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(a2 i1 w))" and
  a1_i2 : "(a1 i2 i2) ∧ (a1 i2 i1) ∧ (a1 i2 i3)
          ∧ (∀w. ((w ≠ i1) ∧ (w ≠ i2) ∧ (w ≠ i3)) ⟶ ¬(a1 i2 w))" and
  a2_i2 : "(a2 i2 i2) ∧ (a2 i2 i3) ∧ (a2 i2 i4)
          ∧ (∀w. ((w ≠ i2) ∧ (w ≠ i3) ∧ (w ≠ i4)) ⟶ ¬(a2 i2 w))" and
  a1_i3 : "(a1 i3 i3) ∧ (a1 i3 i1) ∧ (a1 i3 i2)
          ∧ (∀w. ((w ≠ i1) ∧ (w ≠ i2) ∧ (w ≠ i3)) ⟶ ¬(a1 i3 w))" and
  a2_i3 : "(a2 i3 i3) ∧ (a2 i3 i2) ∧ (a2 i3 i4)
          ∧ (∀w. ((w ≠ i2) ∧ (w ≠ i3) ∧ (w ≠ i4)) ⟶ ¬(a2 i3 w))" and
  a1_i4 : "(a1 i4 i4) ∧ (a1 i4 i5) ∧ (a1 i4 i6)
          ∧ (∀w. ((w ≠ i4) ∧ (w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(a1 i4 w))" and
  a2_i4 : "(a2 i4 i4) ∧ (a2 i4 i2) ∧ (a2 i4 i3)
          ∧ (∀w. ((w ≠ i2) ∧ (w ≠ i3) ∧ (w ≠ i4)) ⟶ ¬(a2 i4 w))" and
  a1_i5 : "(a1 i5 i5) ∧ (a1 i5 i4) ∧ (a1 i5 i6)
          ∧ (∀w. ((w ≠ i4) ∧ (w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(a1 i5 w))" and
  a2_i5 : "(a2 i5 i5) ∧ (a2 i5 i1) ∧ (a2 i5 i6)
          ∧ (∀w. ((w ≠ i1) ∧ (w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(a2 i5 w))" and
  a1_i6 : "(a1 i6 i6) ∧ (a1 i6 i4) ∧ (a1 i6 i5)
          ∧ (∀w. ((w ≠ i4) ∧ (w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(a1 i6 w))" and
  a2_i6 : "(a2 i6 i6) ∧ (a2 i6 i1) ∧ (a2 i6 i5)
          ∧ (∀w. ((w ≠ i1) ∧ (w ≠ i5) ∧ (w ≠ i6)) ⟶ ¬(a2 i6 w))" and
```

```
theorem ¬ ⌊□ (¬ gstit A )⌋ᵢ
```

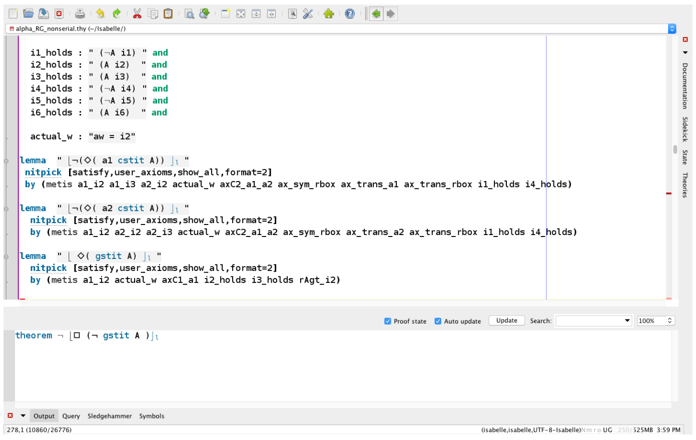
276,61 (10857/26776) (isabelle,isabelle,UTF-8-Isabelle)

Figure 11: Isabelle/HOL embedding for the Group Action Example (part 2)

Finally, we have to define for each world whether it holds the proposition A or $\neg A$ and we have to set the actual world, i.e. the world on which we have evaluated the propositions. In this case, we choose i_2 as the actual world.

In this example, at world i_2, it is not possible for both agents to individually see to it that A, meaning that propositions $\Diamond[a1\ cstit:\ A]$ and $\Diamond[a2\ cstit:\ A]$ are false at i_2. However, by acting together, it is possible that both agents do see to it that A, so $\Diamond[Agt]A$ is true at i_2.

Sledgehammer is able to provide a proof in order to show that the propositions are valid at the world i_2 and when asked for a model that satisfies those propositions, $Nitpick$ came up with the same model as we specified. This is shown in figure 12.

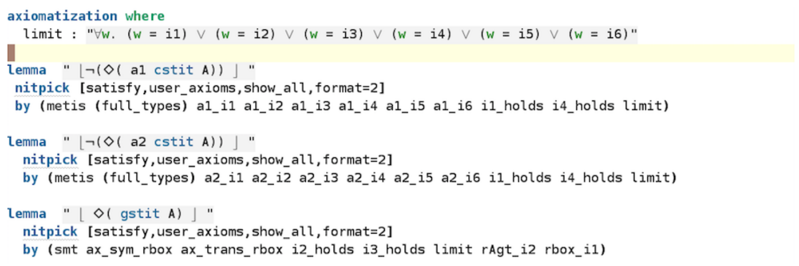


```
  i1_holds : " (¬A i1) " and
  i2_holds : " (A i2)  " and
  i3_holds : " (A i3)  " and
  i4_holds : " (¬A i4) " and
  i5_holds : " (¬A i5) " and
  i6_holds : " (A i6)  " and

  actual_w : "aw = i2"

lemma  " ⌊¬(◇( a1 cstit A)) ⌋ι "
 nitpick [satisfy,user_axioms,show_all,format=2]
 by (metis a1_i2 a1_i3 a2_i2 actual_w axC2_a1_a2 ax_sym_rbox ax_trans_a1 ax_trans_rbox i1_holds i4_holds)

lemma  " ⌊¬(◇( a2 cstit A)) ⌋ι "
  nitpick [satisfy,user_axioms,show_all,format=2]
  by (metis a1_i2 a2_i2 a2_i3 actual_w axC2_a1_a2 ax_sym_rbox ax_trans_a2 ax_trans_rbox i1_holds i4_holds)

lemma  " ⌊ ◇( gstit A) ⌋ι "
  nitpick [satisfy,user_axioms,show_all,format=2]
  by (metis a1_i2 actual_w axC1_a1 i2_holds i3_holds rAgt_i2)
```

```
theorem ¬ ⌊□ (¬ gstit A )⌋ι
```

Figure 12: Isabelle/HOL embedding for the Group Action Example (part 3)

Not only are the propositions valid at the world i_2, they also hold in every world of the model. In order that $Sledgehammer$ is able to find a proof for that, we have to limit the tool so that only considers the 6 possible worlds. This is represented in figure 13.



```
axiomatization where
  limit : "∀w. (w = i1) ∨ (w = i2) ∨ (w = i3) ∨ (w = i4) ∨ (w = i5) ∨ (w = i6)"

lemma  " ⌊¬(◇( a1 cstit A)) ⌋ "
 nitpick [satisfy,user_axioms,show_all,format=2]
 by (metis (full_types) a1_i1 a1_i2 a1_i3 a1_i4 a1_i5 a1_i6 i1_holds i4_holds limit)

lemma  " ⌊¬(◇( a2 cstit A)) ⌋ "
  nitpick [satisfy,user_axioms,show_all,format=2]
  by (metis (full_types) a2_i1 a2_i2 a2_i3 a2_i4 a2_i5 a2_i6 i1_holds i4_holds limit)

lemma  " ⌊ ◇( gstit A) ⌋ "
  nitpick [satisfy,user_axioms,show_all,format=2]
  by (smt ax_sym_rbox ax_trans_rbox i2_holds i3_holds limit rAgt_i2 rbox_i1)
```

Figure 13: Isabelle/HOL embedding for the Group Action Example (part 4)

All the formulas for Horty's example could be proven in Isabelle/HOL. For this particular example, the model was composed only of one single moment and the formulas, which we verified, did not include any temporal operators. However, it would still be interesting to see if Isabelle/HOL is able to evaluate some formulas with some temporal operators for a model. Thus we specified a model from Lorini's paper [22] which he used in order to illustrate the semantics of temporal STIT logic. We already mentioned that temporal Kripke STIT models are infinite and this can also be seen in the model which is depicted in figure 14. The model is composed of five moments but actually, there are infinitely more which is represented by the dotted lines. In Isabelle/HOL we specified a finite version of this model meaning that we only considered the five moments that are depicted in the figure and removed the temporal relation for the 'outer' moments. The formulation of this model was done in a similar way as for the model of the previous example.

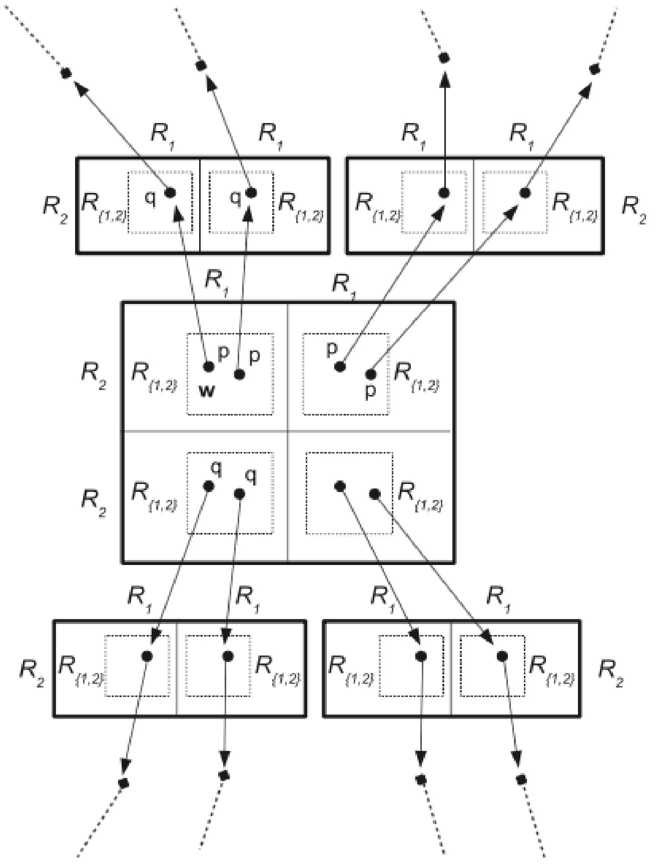
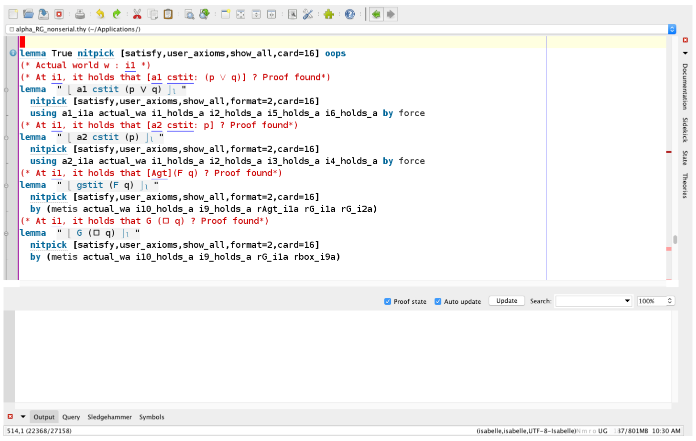


Figure 14: Temporal Kripke STIT model from Lorini's paper

The world w is considered as the actual world and Lorini mentions three formulas which hold for the infinite model, but they also hold for the finite version. For that finite model M, we have that $M, w \vDash [2]p$ since p holds in every world that is in agent 2's choice at world w. For agent 1, we have that $M, w \vDash [1](p \vee q)$ since either p or q holds in every world in agent 1's choice at world w. Further, we have that the group, consisting of agents 1 and 2, sees to it that q will be true at some point in the future.This is formulated as $M, w \vDash [Agt]Fq$. Those three formulas could also be verified by the $Sledgehammer$ tool. For the finite model, we also have that $M, w \vDash G(\Box q)$ which could be proved as well.



alpha_RG_nonserial.thy (~/Applications/)

```
lemma True nitpick [satisfy,user_axioms,show_all,card=16] oops
(* Actual world w : i1 *)
(* At i1, it holds that [a1 cstit: (p ∨ q)] ? Proof found*)
lemma  " ⌊ a1 cstit (p ∨ q) ⌋ι "
  nitpick [satisfy,user_axioms,show_all,format=2,card=16]
  using a1_i1a actual_wa i1_holds_a i2_holds_a i5_holds_a i6_holds_a by force
(* At i1, it holds that [a2 cstit: p] ? Proof found*)
lemma  " ⌊ a2 cstit (p) ⌋ι "
  nitpick [satisfy,user_axioms,show_all,format=2,card=16]
  using a2_i1a actual_wa i1_holds_a i2_holds_a i3_holds_a i4_holds_a by force
(* At i1, it holds that [Agt](F q) ? Proof found*)
lemma  " ⌊ gstit (F q) ⌋ι "
  nitpick [satisfy,user_axioms,show_all,format=2,card=16]
  by (metis actual_wa i10_holds_a i9_holds_a rAgt_i1a rG_i1a rG_i2a)
(* At i1, it holds that G (□ q) ? Proof found*)
lemma  " ⌊ G (□ q) ⌋ι "
  nitpick [satisfy,user_axioms,show_all,format=2,card=16]
  by (metis actual_wa i10_holds_a i9_holds_a rG_i1a rbox_i9a)
```

514,1 (22368/27158) (isabelle,isabelle,UTF-8-Isabelle)Nmro UG 137/801MB 10:30 AM

Figure 15: Isabelle/HOL: Model from Lorini's paper

Bibliography

[1] C. Benzmüller and L. C. Paulson. *Quantified Multimodal Logics in Simple Type Theory*. Logica Universalis (Special Issue on Multimodal Logics), 7(1), pp. 7–20, doi:10.1007/s11787-012-0052-y. 2013.

[2] C. Benzmüller, M. Claus, and N. Sultana. *Systematic verification of the modal logic cube in Isabelle/HOL.* arXiv:1507.08717, 2015.

[3] C. Benzmüller, B. Woltzenlogel Paleo . *Higher-Order Modal Logics: Automation and Applications* 2015

[4] C. Benzmüller, D. Gabbay, V. Genovese and D. Rispoli. *Embedding and Automating Conditional Logics in Classical Higher-Order Logic*. Annals of Mathematics and Artificial Intelligence. 66. 2011.

[5] C. Benzmüller, A. Farjami, and X. Parent. *Faithful semantical embedding of a dyadic deontic logic in HOL*. Technical report, CoRR, 2018. https://arxiv.org/abs/1802.08454.

[6] C. Benzmüller, X. Parent, and L. van der Torre. *A Deontic Logic Reasoning Infrastructure*. 60-69. 10.1007/978-3-319-94418-0_6. 2018.

[7] C. Benzmüller, X. Parent. *I/O Logic in HOL - First Steps*, Computer Science and Communications, University of Luxembourg, Luxembourg. (2018)

[8] N. Belnap, M. Perloff, and M.Xu, *Facing the future: agents and choices in our indeterminist world*, Oxford University Press, New York, 2001.

[9] N. Belnap and M. Perloff. *Seeing to it that: A canonical form for agentives*. In Knowledge Representation and Defeasible Reasoning, H. E. Kyburg, R. P. Loui and G. N. Carlson, eds, pp. 167–190. Kluwer, Boston, 1990.

[10] J. C. Blanchette and T. Nipkow. Nitpick: A counterexample generator for higher-order logic based on a relational model finder. In *International Conference on Interactive Theorem proving*, pages 131-146

[11] J. C. Blanchette and L. C. Paulson. *Hammering Away - A User's Guide to Sledgehammer for Isabelle/HOL*, 2017.

[12] Jan Broerson. *A logical analysis of the interaction between 'obligation-to-do' and 'knowingly doing'*, Deon 2008, vol. 5076, pp. 140-154. Springer, Heidelberg (2008)

[13] Jan Broersen. *A Complete STIT Logic for Knowledge and Action, and Some of Its Applications*. 47-59. 10.1007/978-3-540-93920-7_4. (2008)

[14] Brian F. Chellas, *The Logical Form of Imperatives*, Ph.D. thesis, Philosophy Department, Stanford University, 1969.

[15] Brian F. Chellas, *Time and modality in the logic of agency*, Studia Log- ica 51, no. 3/4, pp. 485–518. 1992.

[16] A. Church. A formulation of the simple theory of types. *In The journal of symbolic logic*, volume 5, pages 56–68. Cambridge Univ Press, 1940.

[17] Henkin, L. A theory of propositional types. *Fundamenta Mathematicae 52* (1963), 323–344

[18] A.Herzig, F. Schwarzentruber, Properties in logics of individual and group agency, *Advances in Modal Logic* (Vol. 7, pp. 133-149), London: King's College, 2008.

[19] John F. Horty, *Agency and Deontic Logic*, Oxford University Press, 2001.

[20] John F. Horty and Nuel Belnap, *The Deliberative Stit: A Study of Action, Omission, Ability, and Obligation*, Journal of Philosophical Logic, Vol. 24, No. 6, pp. 583-644, 1995.

[21] I.L. Humberstone. *Two sorts of 'ought's*. Analysis, 32:8-11. 1971.

[22] E. Lorini. Temporal STIT logic and its application to normative reasoning. *Journal of Applied Non-Classical Logics*, 23(4):372-399.

[23] David Makinson and Leendert van der Torre, *Input/output logics*, Journal of Philosophical Logic, 29(4), 383–408, (2000).

[24] David Makinson and Leendert van der Torre. 2001. *Constraints for input/output logics*. Journal of Philosophical Logic 30, 2 (2001), 155–185.

[25] Thomas Nagel. Moral luck. *In Mortal questions*, 24–38. New York: Cambridge University (1979)

[26] T. Nipkow, L. C. Paulson and M. Wenzel, *Isabelle/HOL - A Proof Assistant for Higher-Order Logic*, volume 2283 of *LNCS*, Springer, 2002.

[27] X. Parent and L. van der Torre. *The pragmatic oddity in a norm-based semantics*. In G. Governatori, editor, Proceedings of the 16th Edition of the International Conference on Artificial Intelligence and Law, ICAIL '17, pages 169–178, New York, NY, USA, 2017. ACM.

[28] Russell, B. Mathematical logic as based on the theory of types. *American Journal of Mathematics 30* (1908), 222–262.

[29] Xin Sun. Input/Output STIT Logic for Normative Systems, Computer Science and Communications, University of Luxembourg, Luxembourg. (2015)

[30] Bernard Williams. Moral Luck. *Moral Luck: Philosophical Papers 1973-1980*. Cambridge: Cambridge University Press. pp. 20–39. (1981)

[31] Ming Xu, *Axioms for deliberative STIT*, Journal of Philosophical Logic, 27, 505–552, (1998)