Game Theory and Logic for Non-repudiation Protocols and Attack Analysis

Matthijs Melissen

Supervisors:

Prof. Dr. S. Mauw (University of Luxembourg)
Prof. Dr. L. van der Torre (University of Luxembourg)
Dr. W. Jamroga (University of Luxembourg)

The author was employed at the University of Luxembourg and received support from the National Research Fund Luxembourg (reference PHD/09/082) in the project “Games for Modelling and Analysis of Security”.

UNIVERSITÉ DU LUXEMBOURG

PhD-FSTC-2013-25
The Faculty of Sciences, Technology and Communication

DISSERTATION

Presented on 14/10/2013 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

by

Matthijs Melissen

Born on 20 June 1985 in Breda (The Netherlands)

Game Theory and Logic for Non-repudiation Protocols and Attack Analysis

Dissertation defense committee

Dr. Wojciech Jamroga, vice-chairman
Université du Luxembourg

Dr. Steve Kremer
Inria Nancy – Grand Est, LORIA

Dr. Fabio Martinelli
IIT-CNR, Italy

Dr. Sjouke Mauw, dissertation supervisor
Professor, Université du Luxembourg

Dr. Leendert van der Torre, chairman
Professor, Université du Luxembourg

Summary

Research in security has traditionally focused on preventing all possible attacks, without focusing on the attacker and defender, and their incentives, abilities, and knowledge. However, the security of a system might depend on such aspects. Therefore, we use logic and artificial intelligence to model the attacker, the defender, and their interaction. We have identified three areas in which the interaction between attacker and defender, and thus the application of game theory and logic, is relevant.

First, we study security protocols. A security protocol specifies the behavior of agents that interact with the aim of guaranteeing security properties. As security protocols are executed by interacting agents with possibly diverging interests, they are an interesting candidate for game-theoretical and logical modeling. We study in particular non-repudiation protocols, which are protocols that guarantee that an agent cannot deny having executed a certain event. We introduce a game-theoretical framework for analysis of security protocols. This framework is created by unifying the Cremers–Mauw protocol semantics with concurrent game structures, which allows us to phrase security properties as formulas of Alternating-time Temporal Logic. We point out two limitations of current game-theoretical approaches, namely dealing with imperfect information and combining fairness and effectiveness, and propose solutions for them. Furthermore, we model the restrictions on the behavior of agents, and the knowledge agents have of such restrictions. If agents lack this knowledge, a new class of attacks, called virtual multi-protocol attacks, arises. Finally, we study the security of security protocols when taking the incentives of agents into account.

Second, we consider farsighted games, which are games in which agents consider future deviations of other agent. Farsighted games are based on the interaction between an initial deviation made by an agent who can be seen as an attacker, and follow-up deviations by the other agents, who can be seen as defenders. We propose a new solution concept based on this idea.

Third, we apply game theory to attack modeling, and more particularly to attack-defense trees. Attack-defense trees are a method to describe possible security weaknesses of a system, and their countermeasures. We make explicit the link between attack–defense trees and a class of extensive-form games.

Acknowledgments

Writing a PhD thesis is an individual effort, but one that involves the support of a large number of people.

I am very grateful to my supervisors. First, I would like to thank Sjouke Mauw for guiding me through my PhD. His research style and his ability to get very quickly to the core of any problem has been a great inspiration for me. I would also like to thank Leon van der Torre, who always encouraged me to see my research in a wider interdisciplinary perspective. I will also miss the regular dinners we had at his home. I am also grateful to Wojtek Jamroga. By working with him, I have gained a large amount of knowledge. Moreover, I have great admiration for his never-ending enthusiasm, and his ability to spontaneously come up with complete lectures in answer to all my questions.

Furthermore, I would like to thank the external members of the thesis defense committee, Steve Kremer and Fabio Martinelli, for assessing this thesis, and for their valuable feedback.

During the process of conducting the research that lead to this thesis, I have discussed my work with many people in my field. In particular, I am grateful for the useful suggestions I received from Cas Cremers, Mohammad Torabi Dashti, Laurent Doyen, Saša Radomirović, Jean-François Raskin, and Henning Schnoor.

In between doing research, I had a great time spending my lunch breaks with my colleagues in Sjouke's research group. I am grateful for the fun and supportive environment, so it is a great pleasure to thank Baptiste Alcalde, Xihui Chen, Ton van Deursen, Naipeng Dong, Hugo Jonker, Simon Kramer, Barbara Kordy, Piotr Kordy, Andrzej Mizera, Tim Muller, Jun Pang, Georgios Pitsilis, Patrick Schweitzer, Rolando Trujillo, Chenyi Zhang, and Yang Zhang.

I would also like to thank my colleagues in Leon's group: Diego Agustín Ambrossio, Guillaume Aucher, Mathijs de Boer, Richard Booth, Patrice Caire, Martin Caminada, Silvano Colombo Tosatto, Dov Gabbay, Valerio Genovese, Llio Humphreys, Xavier Parent, Gabriella Pigozzi, Mikołaj Podlaszewski, Tjitze Rienstra, Francois Schwarzentruber, Marija Slavkovik, Xin Sun, Masoud Tabatabaei, Paolo Turrini, Srdjan Vesic, Emil Weydert, Yining Wu, Marc van Zee, and Pouyan Ziafati.

Furthermore, I would like to thank my parents and siblings for their support. Finally, I would like to thank Natalia, who hardly saw me at home the last months before completing my thesis, for her continuous love, support, and patience.

Matthijs Melissen

Contents

1

Introduction

1.1 Formal Modeling in Security

Security of computer systems plays a critical role in our society. We rely on digital communication methods such as e-mail and internet phone calls, assuming that other people cannot listen in to our conversations. We also use computer systems to keep track of our medical data, and assume that these systems do not allow third parties to access our patient records. Moreover, we assume that nuclear power plants do not explode, even if terrorists try to attack them.

However, these assumptions do not always hold. In May 2013, it was revealed that the National Security Agency of the United States is able to intercept our e-mail messages [Gua13]. In April 2012, log-in details to a Luxembourgish medical system were stolen, which enabled a member of the public to access medical data of thousands of patients in Luxembourg [Wor13]. Nuclear power plants have (so far) not exploded as a consequence of terrorist activity, but in June 2010, a virus was discovered that was suspected to be created by the United States and Israel in order to attack Iran's nuclear facilities [Tim13].

As shown by these examples, we often assume that computer systems have the property that they prevent against a certain type of harm caused by a malicious entity. We call such a property a *security property*, and we call the malicious entity an *attacker*. If the security property does not hold in a system, we say that the system has a *security vulnerability*. Defining security properties, and making sure that they hold in computer systems, is not an easy task. *Computer security* is the field of computer science that deals with defining adequate security properties, and with designing systems in which these properties hold.

Security properties are often not precisely defined. For example, we mentioned that we assume that a medical system does not allow third parties to access patient data. However, this is not a complete specification. It is not clear who are third parties and who are not. Furthermore, it is not specified whether the system is allowed to give access to aggregated or statistical data, instead of data of individual patients. Moreover, we do not know what is actually meant with 'access'.

If a security property is not precisely defined, it might be the case that it is interpreted differently by different people. For example, the designer of a system might think that the security property means one thing, while the user thinks it means something else. In that case, the user's expectation of security turns out to be false.

The imprecise specification of security properties does indeed lead to security problems in real computer systems. An example of this is a vulnerability that

was discovered by the author of this thesis in the trouble ticket system Request Tracker [Pra]. A trouble ticket system is a system that is used in an organization's customer support center to manage reported customer issues. This vulnerability allowed the author to send e-mails that looked like they were sent by a Luxembourgish governmental organization (technically, the author had the possibility to digitally sign arbitrary text with the PGP key of this organization). The underlying cause of this vulnerability was that PGP signing was understood by the receiver of the e-mail to guarantee that a message has been intentionally sent by the signer, while it was understood by the system designer to guarantee that a message originated from the e-mail server of the signer. This lead to a situation where the electronic signatures did not guarantee the property that the user expected, which could be exploited by a malicious entity. To prevent possible abuse of this vulnerability, the governmental organization and the vendor of the software were alerted, and they subsequently resolved the issue. This vulnerability is discussed in more detail in Section 5.3 of this thesis.

Even when the security properties of a system are precisely defined, it might still be the case that they do not hold. Unfortunately, real-life systems in fact often do not satisfy their security properties. Research in web security, carried out by the author of this thesis, has shown that many websites do not satisfy their required security properties [SaT]. Furthermore, operating systems like Microsoft Windows and Ubuntu are forced to release hundreds of security patches per year. Moreover, a well-known security protocol, the *Needham-Schroeder Public-key Protocol*, was thought to be correct for almost twenty years, until an attack was found [Low96]. This shows that many systems used in the real world in fact do not satisfy the security properties they are assumed to satisfy.

The research in this thesis is motivated by the idea that to create secure systems, it is important to first precisely define security properties, and subsequently mathematically prove that a system satisfies these properties. Informal definitions inevitably lead to multiple interpretations, which in turn may lead to security vulnerabilities. Moreover, the large number of vulnerabilities in real-world computer systems shows that it is very difficult to get security right. Mathematical tools can indisputably prove that a system satisfies its required security properties.

1.2 Game Theory and Logic in Security

Security properties are different from other desirable properties of computer systems. For most desirable properties of computer systems, it is the case that if the system does not satisfy the property, everyone is worse off. It is in nobody's advantage if a program displays text in bold when the button for italics is selected. If a computer-controlled elevator does not arrive when it is called, neither the building owner nor the user of the elevator will be happy. If a medical system stops working, both the patient and the doctor are in trouble.

For a security property, however, there exists an entity, the attacker, that profits from the fact that the property does not hold. Consider, for instance, an e-mail system. E-mail systems should have the property that people cannot intercept e-mails. If this property does not hold in the system, a spy might actually be happy

if he can listen in. Similarly, an enemy nation might like to be able to gain access to the control of a nuclear plant.

To model security properties, we should not only consider the system itself, but also who the attacker is, and what his incentives, abilities and knowledge are. For example, if the attacker has no incentive to attack the system, because the attack costs the attacker more than he profits from it, then we might still consider the system secure. Alternatively, a system could be seen as secure if the attacker lacks the ability to attack the system. Similarly, we could consider a system secure if the attacker does not have the required knowledge to attack the system.

In addition, we must take into account the entity that is under attack, and its interaction with the attacker. We call the entity that is under attack the *defender*. It might be that the defender has no incentive to prevent an attack, because the cost of preventing the attack is higher than the damage caused by the attack. It might also be that the attacker can at first attack the system, but that the defender has a way to respond and prevent the attack. Furthermore, it might be that when there exists a way to defend against an attack, the defender does not have the ability to fend off this attack, for example because he has insufficient information. All these circumstances are relevant for determining whether a system is secure or not.

Therefore, when we model security properties, we need to model the attacker, the defender, and their interaction. In this thesis, we model the attacker and the defender as *agents*. Agents are autonomous entities, typically with diverging information or diverging interests, or both [SLB09]. They are typically software modules or pieces of hardware, but the ideas apply to humans or even entire organizations as well. We use *game theory* and *logic* to describe and predict the behavior of agents.

Game theory helps us to understand the phenomena we observe when rational agents interact [OR94]. We illustrate this by considering an extremely simple protocol as a game. We assume that two people who are operating in the same business, named Alice and Bob, simultaneously choose to send either a piece of useful information, or a piece of useless information. Alice and Bob expect that their business will gain 5000 euro from receiving useful information from the other person. As Alice and Bob are competitors, they prefer however not to give out useful information. They estimate that it will cost their business 1000 euro to give useful information to the other person. Therefore, if both agents send useful information, they each gain $5000 - 1000 = 4000$ euro. If both agents send useless information, they each gain or lose nothing. If one person sends useful information while the other person sends useless information, the person sending useful information loses 1000 euro, and the person receiving useful information gains 4000 euro. We can model this situation as a *game*. We represent this game as a table:

	Useful	Useless
Useful	$(4000, 4000)$	$(-1000, 5000)$
Useless	$(5000, -1000)$	$(0, 0)$

This table shows that both agents have two *strategies*: sending useful information, or sending useless information. The rows represent Alice's strategies, and the columns represent Bob's strategies. The combination of a strategy of Alice and a

strategy of Bob is called a *strategy profile*. Each cell represents the *utilities* of a strategy profile for both agents. The first component of the pair represents Alice's utility, and the second component represents Bob's utility.

Let us consider what Alice should do. We can see that if Bob sends useless information, it is best for Alice to also send useless information, because that gives her a utility of 0 instead of -1000 euro. If Bob sends useful information instead, it is still better for Alice to send useless information, as that gives her a utility of 5000 euro instead of 4000 euro. In other words, whatever Bob does, it is best for Alice to send useless information. Therefore, game theory predicts that if two rational agents execute this protocol, both of them will send useless information. This is true despite the fact that both agents would be better off had they both chosen to send the piece of useful information.

To model the agents, we use *logic*. A logic is concerned with valid reasoning. A variety of logics have been developed for different purposes. We focus in particular on logics developed in the field of *multi-agent system*. Multi-agent systems is a field originating from artificial intelligence. Multi-agent systems are those systems that include multiple autonomous entities with either diverging information or diverging interests, or both [SLB09]. Agents might be software or hardware agents living on the internet, but the ideas apply to humans as well. Logics have turned out to be a useful tool to model the knowledge, beliefs, desires, intentions and abilities of agents. For example, *epistemic logic* has been developed to reason about the knowledge of agents (see e.g. [FHMV95]). Furthermore, to reason about believes, desires, and intentions, the BDI model was proposed [Bra87]. Moreover, for reasoning about the abilities of agents, logics such as *Alternating-time Temporal Logic* [AHK02] and Strategy Logic [CHP10] have been developed.

Game theory and logic are in general tools designed to model the incentives, abilities, and knowledge of agents. It can therefore be expected that they can serve a useful role in modeling the attacker and defender, and consequentially in modeling security protocols.

1.3 Non-repudiation

In this thesis, we repeatedly refer to a security property which is called *non-repudiation* [ZG96]. Non-repudiation is, basically, the requirement that an agent cannot deny having executed a certain event. We will illustrate this security property on the basis of two scenarios.

Consider a teacher who wants to send the marks of a course to his students by e-mail. A complication is that it is possible for a malicious person to generate an e-mail that appears to be coming from someone else. This is because the e-mail protocol does not verify that the indicated sender is indeed the real sender. Therefore, the student who receives the e-mail does not know whether the e-mail is sent by the teacher, or spoofed by another student playing a prank on him. Moreover, if the university administration does not believe that the e-mail has been sent by the teacher, the student has no evidence to convince the administration otherwise. The desired property in this scenario is *non-repudiation of origin*, which means that the sender should not be able to deny having sent a message.

Now we consider a customer of a mobile phone company who wants to cancel his contract. The customer sends a cancellation letter by e-mail. The customer does not know whether the mobile phone company has received the cancellation. It would be desirable that the company cannot deny having received the cancellation. However, the customer has no way to prove that the company has received the letter, so the company can deny having received it. It does not help if the company confirms the cancellation by e-mail, as the company can also deny having sent the confirmation. The desired property in this scenario is *non-repudiation of receipt*, which means that the receiver should not be able to deny having received a message.

Protocols that achieve non-repudiation work by making a protocol participant collect evidence, which in case of disagreement does not only convince himself, but also a third party, called the judge. In the student–teacher scenario, the administration acts as a judge. In the case of the mobile phone company, the local court would act as a judge.

In non-repudiation protocols, it is often required to have *fair exchange* of evidences. Fair exchange is a security property that guarantees that two items get exchanged fairly. Fair exchange of evidences means, basically, that if one party is able to get evidence of origin, the other party is able to get evidence of receipt, and vice versa.

We focus in this thesis on non-repudiation protocols, since they have an interesting property that many other properties lack. Non-repudiation is mostly concerned with dishonest behavior of the participants in the protocol itself, while other security properties are typically concerned with external attackers. For instance, in a protocol satisfying non-repudiation of receipt of a message, the receiving agent may attack by trying to obtain the message without providing evidence of receipt, and the sender may attack by trying to obtain evidence of receipt without actually providing the message.

Therefore, in non-repudiation protocols, protocol participants have both an incentive to cooperate, as they need to run the protocol together in order to get evidence, and an incentive to compete, as they try to get more pieces of evidence than they are supposed to. The tension between the incentive to cooperate and the incentive to compete make such protocols interesting objects to study using game theory and logic.

1.4 Research Question

We have seen that in order to define security properties, it is important to model the attacker, the defender, and their interaction. Game theory and logic are tools to model agents and their interaction. We therefore expect to gain new insights by using game theory and logic for modeling the interaction between an attacker and a defender. This brings us to our research question.

Research question: *How can game theory and logic help us to model the interaction between an attacker and a defender?*

This question is clearly very broad, as the interaction between attacker and defender plays a role in many fields in security. We therefore identify three areas in which we believe that the interaction between attacker and defender is relevant,

and restrict ourselves to these areas. These areas are *security protocols*, *farsighted games*, and *attack modeling*. Each of these areas leads to a subquestion.

Subquestion 1. How can game theory and logic help us to model the interaction between an attacker and a defender in security protocols?

A *security protocol* specifies the behavior of interacting agents, with the aim of guaranteeing security properties for the participating agents. As security protocols are executed by interacting agents with possibly diverging interests, we expect that they are an interesting candidate for game-theoretical and logical modeling. We analyze four different aspects of security properties.

First, we investigate how security protocols themselves can be defined in a game-theoretical model. This model creates the basis for analyzing the security properties we consider when studying the other aspects.

We proceed by modeling the *information* protocol participants have. When modeling security protocols, it is usually assumed that agents have *perfect information*, i.e., they know exactly the global state of the system, including the local states of all other agents. This is an unrealistic assumption. Therefore, to get a more realistic model, we investigate how we can model imperfect information of security properties.

If agents do not know how other agents behave, they cannot make any claims about such agents [KSW97]. Therefore, it is important to model the restrictions on the behavior of agents, and the knowledge agents have of such restrictions. The third part of our analysis is concerned with modeling this aspect.

The fourth aspect concerns the incentives of the protocol participants. It is usually assumed that protocols need to protect against all attacks, including attacks in which agents act against their own incentives. By making assumptions about the incentives of agents, and predicting the behavior of these agents based on their incentives, we might accept more protocols as secure. Game theory is designed to predict how agents behave given their incentives. Therefore, we investigate how game theory can be used to model this interaction.

Subquestion 2. How can game theory and logic help us to model the interaction between an attacker and a defender in farsighted games?

A *solution concept* is a formal rule that predicts how agents behave in a game based on their utilities. Some well-known solution concepts, such as Nash equilibrium (e.g. [OR94]), predict that agents will end up playing a strategy profile where no agent has an incentive to deviate to another strategy profile. Nash equilibrium does not take into account that agents might be 'farsighted', i.e., take future deviations into account. Farsighted games are based on the interaction between an initial deviation made by an agent who can be seen as an attacker, and follow-up deviations by the other agents, who can be seen as defenders. We investigate how we can define a solution concept in farsighted games. Moreover, we investigate how the interaction between an attacker and a defender can be modeled as a game on a meta-level.

Subquestion 3. How can game theory and logic help us to model the interaction between an attacker and a defender in the field of attack modeling?

Attack modeling is concerned with modeling all possible attacks on a system. *At-*

tack trees [Sch99] are a method to describe possible security weaknesses of a system. Recently, it has been proposed to extend attack trees with countermeasures taken by the defender, resulting in attack-defense trees [KMRS10]. In attack-defense trees, the interaction between an attacker and a defender therefore plays an important role. This interaction reminds us of the interaction between two agents in extensive-form games. We therefore try to make explicit the connection between attack-defense trees and extensive-form games.

1.5 Methodology

The general methodology used in this thesis is characterized by an interdisciplinary approach. To analyze computer security, we rely on techniques developed in game theory, logic, and multi-agent systems. We will discuss in more detail the methodology we use to answer each of the three individual subquestions.

Subquestion 1. How can game theory and logic help us to model the interaction between an attacker and a defender in security protocols?

We model security protocols using a variant of the Cremers–Mauw protocol semantics [CM03, CM12]. We keep the original syntax of the protocol definitions. However, instead of interpreting protocols in labeled transition systems, we interpret them in concurrent game structures [AHK02]. This allows us to use logics for reasoning about strategic ability, such as Alternating-time Temporal Logic [AHK02] and Strategy Logic [CHP10], in order to specify security properties. Our approach differs from the game-based protocol specification used by Kremer and Raskin [KR03], in the sense that we allow for agents that execute an unbounded number of protocol roles (albeit at the cost of decidability).

In order to model agents with imperfect information, we use a version of concurrent game structures with imperfect information as developed by Schobbens [Sch04b]. We use Alternating-time Temporal Logic and Strategy Logic to specify our security properties. We interpret these logics in imperfect-information models by using the semantics of Schobbens [Sch04b], and we define our own semantics to interpret Strategy Logic in such models. We apply our modeling by using it to study the specification of fair exchange.

In order to formally model the knowledge that agents need to have about other agents, we use epistemic logic. We test our method by analyzing the components of which the security property of non-repudiation exists. For each of these components, we investigate which knowledge assumptions should hold.

To model the incentives of agents taking part in a security protocol, we use classical game theory. We apply our method to simple examples of non-repudiation protocols.

Subquestion 2. How can game theory and logic help us to model the interaction between an attacker and a defender in farsighted games?

We use the interaction between attacker and defender to define a new solution concept, called *farsighted pre-equilibria*. We study the computational complexity of the solution concept. We also investigate how farsighted pre-equilibria correspond to solving a meta-game obtained by using the original payoff matrix as arena and

the deviations as moves.

Subquestion 3. How can game theory and logic help us to model the interaction between an attacker and a defender in the field of attack modeling?

We use extensive-form games [OR94] to model attack–defense trees. To be more precise, we compare attack–defense trees to two-agent binary zero-sum extensive-form games. We propose a mapping between these two concepts, and show that this mapping is correct by proving that it preserves satisfiability.

1.6 Thesis Overview

We now discuss the structure of this thesis. We analyze security properties (Subquestion 1) in Chapters 2–5. In Chapter 7 , we analyze farsighted games (Subquestion 2). In Chapter 8, we analyze attack modeling. We proceed with discussing the content of the individual chapters.

- **Chapter 2: Modeling Security Protocols as Games**

  We introduce the game-theoretical framework for analysis of security protocols that we use in this thesis. This framework is created by uniting the Cremers–Mauw protocol semantics [CM03, CM12] with concurrent game structures [AHK02].

- **Chapter 3: Imperfect Information in Fair Non-repudiation Protocols**

  We study the verification of fairness in non-repudiation protocols by means of Alternating-time Temporal Logic, and we point out a number of limitations of this approach. The first limitation has to do with the lack of modeling of imperfect information. The second limitation has to do with the fact that fairness is not a sufficient property for fair-exchange protocols, as protocols are also required to be effective. We propose a solution to overcome both of these limitations in isolation. Moreover, we give a hierarchy of the various definitions of fairness, and prove that this hierarchy is correct.

  This chapter is based on joint work with Wojciech Jamroga and Sjouke Mauw [JMM12].

- **Chapter 4: The Expressive Power of** ATL*

  We show that Alternating-time Temporal Logic is not expressive enough to overcome both limitations from Chapter 3 at the same time. Instead, we model fair exchange in Strategy Logic. We do so by giving a characterization of the class of formulas of Alternating-time Temporal Logic that cannot be expressed in Strategy Logic, and showing that the property that expresses fair exchange falls into this class. We also show that in perfect-information models, at least part of the fragment of Strategy Logic can be expressed in Alternating-time Temporal Logic.

  This chapter is based on joint work with Leon van der Torre (to be submitted for publication).

- **Chapter 5: Non-repudiation and Virtual Multi-Protocol Attacks**

  We formally model the property of non-repudiation. Moreover, we introduce a new class of attacks, which we call virtual multi-protocol attacks, and which are, basically, situations in which a security property holds, but in which the security property is not known to hold by some agent. To prevent such attacks, we show that we need an additional assumption on the knowledge that agents have about the protocols that other agents execute.

  This chapter is based on joint work with Sjouke Mauw (submitted for publication).

- **Chapter 6: Incentives in Security Protocols**

  We propose a methodological framework for analyzing security protocols and other interaction protocols that takes into account the incentives of agents. Our idea is based on a set of *defenders* of the protocol that are in favor of the objective of the protocol. We say that a protocol can be defended if it is correct with respect to every utility profile in which the preferences of all defenders comply with the objective. We obtain characterization results for defendability under Nash equilibria.

  This chapter is based on joint work with Wojciech Jamroga and Henning Schnoor [JMS13].

- **Chapter 7: Farsighted Pre-equilibria**

  We propose a new solution concept, which we call farsighted pre-equilibrium. This solution concept takes into account only deviations that do not lead to a decrease of the agent's utility even if some other deviations follow.

  The idea is to "broaden" Nash equilibrium in a way that does not discriminate solutions that look intuitively appealing but are ruled out by Nash equilibrium. Then Nash equilibrium may be interpreted as a specification of play which is certainly rational, and strategy profiles that are not farsighted pre-equilibria can be considered certainly irrational.

  This chapter is based on joint work with Wojciech Jamroga [JM11].

- **Chapter 8: Relating Attack-Defense Trees and Games**

  We make the link between attack–defense trees and a class of extensive-form games explicit. In particular, we show that attack–defense trees and binary zero-sum two-agent extensive-form games have equivalent expressive power when considering satisfiability, in the sense that they can be translated to each other while preserving their outcome.

  This chapter is based on joint work with Barbara Kordy, Sjouke Mauw, and Patrick Schweitzer [KMMS10].

- **Chapter 9: Conclusion and Future Work**

  We draw conclusions, summarize the contributions of this thesis, and point out suggestions for future work.

2

Modeling Security Protocols as Games

Abstract. We develop a game-based protocol semantics, by modeling protocols as concurrent game structures. To be precise, we take protocol specifications from the Cremers–Mauw protocol semantics, and interpret such protocol specifications in concurrent game structures instead of in labeled transition systems. This allows us to define security properties on Cremers–Mauw protocol specifications in logics of strategic ability, such as Alternating-time Temporal Logic.

2.1 Introduction

When studying security protocols, we are interested in the security properties that hold in such protocols. In order to be able to study such properties, we formally introduce security protocols and their semantics. The treatment is based on the operational semantics framework developed by Cremers and Mauw [CM03, CM12], but the definitions and observations in this chapter easily transfer to other frameworks, such as Strand Spaces [THG99].

Our model is different from the treatment by Cremers and Mauw in four ways.

- To allow analysis with Alternating-time Temporal Logic, we use *concurrent game structures* [AHK02] instead of labeled transition systems to describe the protocol interaction.

- The protocol semantics framework by Cremers and Mauw only allows for linear protocols, i.e., protocols that consist of the sequential composition of events. Our treatment follows Van Deursen [VD11] in that we allow for branching protocols by including parallel composition and choice points in our semantics. Moreover, we allow for loops by defining *no-exit iteration* and the *binary Kleene star* operation [BBP94].

- The protocol semantics by Cremers and Mauw only considers finite executions and finite traces. However, this is not sufficient when considering liveness properties, i.e., properties that require that something eventually happens. Such properties are needed, for example, when defining fair exchange. Therefore, we also consider infinite executions and infinite traces.

- For simplicity, we do not consider arbitrary functions, and only define hashing, encryption, and constants.

2.2 Concurrent Game Structures

We model protocols as *concurrent game structures* [AHK02]. An infinite concurrent game structure describes how actions taken by a set of agents can cause a transition from one state to the next. Every state is labeled with a set of propositions. A state and a combination of actions, one for each agent, together define the next state of the system.

We first define *infinite concurrent game structures.*

Definition 2.1 (Infinite concurrent game structures)**.** *An infinite concurrent game structure is a tuple* $M = (\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta)$ *with the following components:*

- *a countable set* $\mathbb{A}\mathrm{gt} = \{a_1, a_2, \ldots\}$ *of* agents*;*
- *a countable set* Act *of* actions*;*
- *a countable set* Q *of* states*;*
- *a countable set* Π *of* propositions*;*
- *a* labeling function π *assigning a set* $\pi(q) \subseteq \Pi$ *of propositions true at* q *to each state* $q \in Q$*;*
- *a set* $d = \{d_1, d_2, \ldots\}$*, such that for each agent* $a_i \in \mathbb{A}\mathrm{gt}$*,* d_i *is a* move function *assigning a set* $d_i(q) \subseteq Act$ *of actions available at state* $q \in Q$ *(we use* $D(q)$ *to denote the set of* action profiles $d_1(q) \times d_2(q) \times \ldots$*);*
- *a* transition function δ *assigning a new state* $\delta(q, (\alpha_1, \alpha_2, \ldots)) \in Q$ *to every combination of state* $q \in Q$ *and action profile* $(\alpha_1, \alpha_2, \ldots) \in D(q)$*.*

We say that a concurrent game structure is finite when the sets of agents, actions, states and propositions are finite.

Definition 2.2 ((Finite) concurrent game structures)**.** *A (finite) concurrent game structure is an infinite concurrent game structure* $M = (\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta)$ *where* $\mathbb{A}\mathrm{gt}$*,* Act*,* Q*, and* Π *are finite.*

Example 2.1. *Consider concurrent game structure* M *(Figure 2.1). It is formally defined as follows.*

$M = (\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d_1, \delta_1)$ *with* $\mathbb{A}\mathrm{gt} = \{a, b\}$*,* $Act = \{1, 2\}$*,* $Q = \{q_0, q_1, q_2, q_3\}$*,* $\Pi = \{p_a, p_b, p_c\}$*,* $\pi = \{q_0 \mapsto \emptyset, q_1 \mapsto \{p_a, p_b, p_c\}, q_2 \mapsto \{p_a, p_b\}, q_3 \mapsto \emptyset\}$*,* $d_1(q_0) = d_2(q_0) = \{1, 2\}$*,* $d_1(q_1) = d_1(q_2) = d_1(q_3) = d_2(q_1) = d_2(q_2) = d_2(q_3) = \{1\}$*,* $\delta(q_0, (1, 1)) = q_1$*,* $\delta(q_0, (2, 1)) = \delta(q_0, (2, 2)) = q_2$*,* $\delta(q_0, (1, 2)) = q_3$*, and finally* $\delta(q_i, (1, 1)) = q_i$ *for* $i \in \{1, 2, 3\}$*.*

In this concurrent game structure, there are two agents, a *and* b*. There are also two actions, 1 and 2. There are four states,* q_0*,* q_1*,* q_2*, and* q_3*. There are three propositions,* p_a*,* p_b*, and* p_c*. In* q_0 *and* q_3*, none of the propositions are true, while all propositions are true in* q_1*, and proposition* p_a *and* p_b *are true in* q_2*. In* q_0*, both players can play 1 and 2, while in* q_1*, they can only play 1. From* q_0*, if both players play 1, they end up in* q_1*. If* a *plays 2, they end up in* q_2 *independent of what* b *plays. Otherwise, they end up in* q_3*. Once in one of these states, they remain there.*

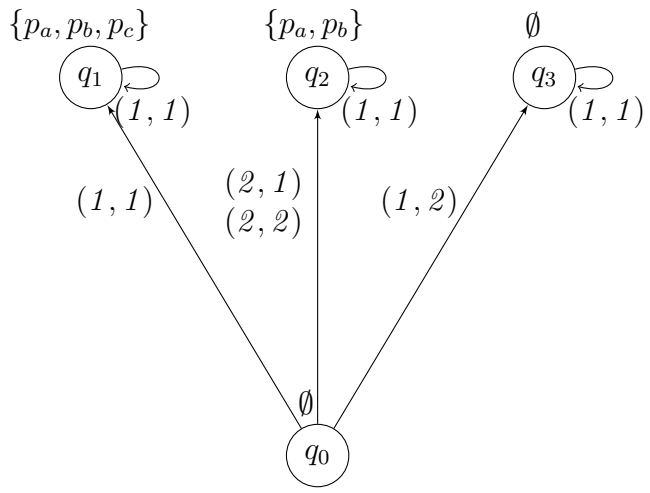


Figure 2.1: An example of a concurrent game structure

A *path* is an infinite sequence of states such that there is a transition between each two subsequent states.

Definition 2.3 (Paths [AHK02])**.** *Let $M = (\mathbb{A}\text{gt}, Act, Q, \Pi, \pi, d, \delta)$ be a concurrent game structure. A path in M is an infinite sequence $\lambda = q_0, q_1, q_2, \ldots$ of states such that for all positions $i \geq 0$, we have $q_{i+1} = \delta(q_i, (\alpha_1, \alpha_2, \ldots))$ for some action profile $(\alpha_1, \alpha_2, \ldots) \in D(q_i)$.*

Now we define the *finite prefix* of a sequence, the *infinite suffix* of a sequence, and the *composition* of two sequences.

Definition 2.4 (Finite prefix, Infinite suffix, Composition)**.** *Given a path $\lambda = q_0, q_1, \ldots$ and a position $i \geq 0$, we use $\lambda[i]$, $\lambda[0, i]$, and $\lambda[i, \infty]$ to denote the state q_i, the finite prefix $q_0, q_1, \ldots, q_i$, and the infinite suffix $q_i, q_{i+1}, \ldots$, respectively. Furthermore, given a finite sequence $\lambda = q_0, \ldots, q_n$ and a sequence $\lambda' = q'_0, \ldots$, the* composition *of λ and λ', denoted $\lambda \cdot \lambda'$, is $q_0, \ldots, q_n, q'_0, \ldots$.*

A strategy of an agent is a function that assigns an action to every finite non-empty sequence of states.

Definition 2.5 (Strategies [AHK02])**.** *Let $M = (\mathbb{A}\text{gt}, Act, Q, \Pi, \pi, d, \delta)$ be a concurrent game structure. We write Q^+ for the set of finite non-empty sequences, i.e., sequences of the form $q_1, \ldots, q_n$ such that $q_i \in Q$ for $1 \leq i \leq n$.*

A strategy *σ_a of agent a is a function from Q^+ to Act such that if $\sigma_a(q_1, \ldots, q_n) = \alpha$, then $\alpha \in d_a(q_n)$. Given a sequence of agents C, a* collective strategy *σ_C for C is a sequence of strategies σ_a, one for each agent $a \in C$.*

We write Σ_a for the set of all strategies of agent a.

A set of agents, an initial state, and a set of strategies together define an outcome, i.e., the set of paths that this set of agents can enforce.

Definition 2.6 (Outcome [AHK02])**.** *Let $M = (\mathbb{A}\text{gt}, Act, Q, \Pi, \pi, d, \delta)$ be a concurrent game structure where $\mathbb{A}\text{gt} = \{a_1, a_2, \ldots\}$, let $q \in Q$ be a state, let $C \subseteq \mathbb{A}\text{gt}$*

be a set of agents, and let $\sigma_C = (\sigma_{b_1}, \sigma_{b_2}, \ldots)$ *be a collective strategy for* C*. The outcome of* σ_C *from state* q*, written as* $out(q, \sigma_C)$*, is the set of paths* $\lambda = q_0, \ldots, q_n$ *such that* $q_0 = q$ *and for all* $k \geq 0$*, there exists* $(\alpha_1, \alpha_2, \ldots) \in D(q_k)$ *such that*

1. $\alpha_i = \sigma_i(\lambda[0, k])$ *for all* i *such that* $a_i \in C$*, and*
2. $\delta(q_k, (\alpha_1, \alpha_2, \ldots)) = q_{k+1}$.

Given an agent a*, sometimes we write* $out(q, \sigma_a)$ *instead of* $out(q, (\sigma_{\{a\}}))$.

A special case of concurrent game structures are *turn-based game structures*. These are concurrent game structures in which at every state, there is one agent determining the next state. If in a state, agent a is solely determining the next state, we call that state an *a-state*.

Definition 2.7 (a-state)**.** *Let* $M = (\mathbb{A}\text{gt}, Act, Q, \Pi, \pi, d, \delta)$ *with* $\mathbb{A}\text{gt} = \{a_1, a_2, \ldots\}$ *be a concurrent game structure, let* $q \in Q$ *be a state, and let* $a_i \in \mathbb{A}\text{gt}$ *be an agent. Then we say that* q *is an* a_i*-state if there exists an action* α_i *such that for all actions* $\alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots$ *and actions* $\beta_1, \ldots, \beta_{i-1}, \beta_{i+1}, \ldots$*, it holds that* $\delta(q, (\alpha_1, \ldots, \alpha_{i-1}, \alpha_i, \alpha_{i-1}, \ldots)) = \delta(q, (\beta_1, \ldots, \beta_{i-1}, \alpha_i, \beta_{i+1}, \ldots))$.

A concurrent game structure is turn-based if every state is an a-state for some agent a.

Definition 2.8 (Turn-based game structure)**.** *We say that a concurrent game structure* $M = (\mathbb{A}\text{gt}, Act, Q, \Pi, \pi, d, \delta)$ *is a turn-based game structure if for every state* $q \in Q$*, there exists an agent* a *such that* q *is an* a*-state.*

2.3 Modeling Security Protocols

In practice, describing security protocols by fully giving the concurrent game structures is cumbersome, as the state space is exponential in the number of rounds of the protocol. Instead, we sometimes specify security protocols by means of the Cremers–Mauw protocol semantics [CM03, CM12].

The formalization of security protocols in the Cremers–Mauw protocol semantics is based on the two notions of *roles* and *runs*. A protocol consists of a collection of roles. Roles might contain uninstantiated variables and role names to which no agents have been assigned. In the execution of a protocol, every role might give rise to one or more runs. A run is modeled as a role together with a unique identifier, an instantiation of the variables, and an assignment of agents to roles.

2.3.1 Role Terms

We start by defining the set of *role terms*. Role terms can be used in the specification of a role, for example as terms that are sent or received. We first consider the basic term sets, such as freshly generated values, roles and variables, which can be used to construct role terms. Afterwards, we add constructors for pairing and cryptographic primitives to construct the set of all role terms.

Definition 2.9 (Basic Term Sets)**.** *We assume the following basic term sets:*

- *Var, denoting* variables *that are used to store received messages;*
- *Const, denoting* constants*, such as natural numbers and strings;*
- *Fresh, denoting* nonces *(numbers used once), i.e., values that are freshly generated for each instantiation of a role;*
- *Role, denoting* roles*.*

The following table contains typical elements for each of the sets defined above.

Description	Set	Typical element
Variables	*Var*	V, W
Constants	*Const*	1, hello
Nonces	*Fresh*	ni, nr
Roles	*Role*	R, S

The basic term sets can be used to construct more complex role terms. We use (t_1, t_2) to denote the *pairing* of role terms t_1 and t_2. Moreover, we define the following *cryptographic primitives*. We write $\mathsf{h}(t)$ for the *hashing* of role term t, and $\{\!|\, t_1 \,|\!\}_{t_2}$ for the *encryption* of role term t_1 with role term t_2. Finally, we write $\mathsf{sk}(R)$ for the *secret key* of role R, $\mathsf{pk}(R)$ for the *public key* of role R, and $\mathsf{k}(R, S)$ for the key *shared* between role R and S.

Definition 2.10 (Role Terms)**.** *The set of role terms RoleTerm is defined as follows.*

$$\begin{aligned} \mathit{RoleTerm} ::= \quad & \mathit{Var} \mid \mathit{Const} \mid \mathit{Fresh} \mid \mathit{Role} \\ \mid \quad & (\mathit{RoleTerm}, \mathit{RoleTerm}) \\ \mid \quad & \mathsf{h}(\mathit{RoleTerm}) \\ \mid \quad & \{\!|\, \mathit{RoleTerm} \,|\!\}_{\mathit{RoleTerm}} \\ \mid \quad & \mathsf{sk}(\mathit{Role}) \mid \mathsf{pk}(\mathit{Role}) \mid \mathsf{k}(\mathit{Role}, \mathit{Role}). \end{aligned}$$

We write RoleTerm for the set of all role terms.

For readability, we sometimes write (t_1, t_2, t_3) *instead of* $((t_1, t_2), t_3)$*, and* $\{\!|\, t_1, t_2 \,|\!\}_k$ *instead of* $\{\!|\, (t_1, t_2) \,|\!\}_k$*.*

We proceed by defining the set of *variables in a role term* and the set of *roles in a role term.*

Definition 2.11 (Variables in a role term)**.** *The set of variables in a role term rt,*

written $vars(rt)$, is defined as follows.

$$vars(rt) = \begin{cases} \emptyset & \text{if } rt \in Fresh \cup Role \cup Const, \\ \{rt\} & \text{if } rt \in Var, \\ vars(rt_1) \cup vars(rt_2) & \text{if } rt = (rt_1, rt_2), \\ vars(rt_1) & \text{if } rt = \mathsf{h}(rt_1), \\ vars(rt_1) \cup vars(rt_2) & \text{if } rt = \{\!|\, rt_1 \,|\!\}_{rt_2}, \\ vars(rt_1) & \text{if } rt = \mathsf{sk}(rt_1), \\ vars(rt_1) & \text{if } rt = \mathsf{pk}(rt_1), \\ vars(rt_1) \cup vars(rt_2) & \text{if } rt = \mathsf{k}(rt_1, rt_2). \end{cases}$$

Definition 2.12 (Roles in a role term)**.** *The set of roles in a role term rt, written $roles(rt)$, is defined as follows.*

$$roles(rt) = \begin{cases} \emptyset & \text{if } rt \in Fresh \cup Var \cup Const, \\ \{rt\} & \text{if } rt \in Role, \\ roles(rt_1) \cup roles(rt_2) & \text{if } rt = (rt_1, rt_2), \\ roles(rt_1) & \text{if } rt = \mathsf{h}(rt_1), \\ roles(rt_1) \cup roles(rt_2) & \text{if } rt = \{\!|\, rt_1 \,|\!\}_{rt_2}, \\ roles(rt_1) & \text{if } rt = \mathsf{sk}(rt_1), \\ roles(rt_1) & \text{if } rt = \mathsf{pk}(rt_1), \\ roles(rt_1) \cup roles(rt_2) & \text{if } rt = \mathsf{k}(rt_1, rt_2). \end{cases}$$

A role term that is encrypted with a key can only be decrypted with the *inverse* of this key. Secret keys and public keys are each other's inverse, enabling asymmetric encryption. All other role terms are their own inverse, enabling symmetric encryption.

Definition 2.13 (Inverse)**.** *We define the inverse function $\ldots^{-1} : RoleTerm \to RoleTerm$ as follows. Given $R \in Role$, we set $\mathsf{pk}(R)^{-1} = \mathsf{sk}(R)$ and $\mathsf{sk}(R)^{-1} = \mathsf{pk}(R)$. Moreover, we set $t^{-1} = t$ for all terms t that are not of the form $\mathsf{sk}(R)$ or $\mathsf{pk}(R)$.*

Given a set of role terms $\mathcal{T}$ and role term t, the term inference relation $\mathcal{T} \vdash t$ expresses that t is derivable from $\mathcal{T}$. This relation is defined as follows.

Definition 2.14 (Term Inference Relation)**.** *Let $\mathcal{T}$ be a set of terms. The term inference relation $\vdash : \mathcal{P}(RoleTerm) \to RoleTerm$ is defined as the smallest relation that satisfies the following properties for all terms $t, t', k \in RoleTerm$.*

$$\begin{aligned} t \in \mathcal{T} &\Rightarrow \mathcal{T} \vdash t; \\ \mathcal{T} \vdash t \wedge \mathcal{T} \vdash t' &\Rightarrow \mathcal{T} \vdash (t, t'); \\ \mathcal{T} \vdash (t, t') &\Rightarrow \mathcal{T} \vdash t \wedge \mathcal{T} \vdash t'; \\ \mathcal{T} \vdash t &\Rightarrow \mathcal{T} \vdash h(t); \\ \mathcal{T} \vdash t \wedge \mathcal{T} \vdash k &\Rightarrow \mathcal{T} \vdash \{\!|\, t \,|\!\}_k; \\ \mathcal{T} \vdash \{\!|\, t \,|\!\}_k \wedge \mathcal{T} \vdash k^{-1} &\Rightarrow \mathcal{T} \vdash t. \end{aligned}$$

The definition states that a term t is derivable from $\mathcal{T}$ if t is an element of $\mathcal{T}$. Furthermore, a pair is derivable if both elements of the pair are derivable, and vice versa. The hash of a term is derivable if the term is derivable (but not vice versa). If a term t and a key k are derivable, then the encryption of t with k is derivable as well. If the encryption of t with k is derivable, then t is derivable whenever the inverse of k is derivable.

Example 2.2. *Let $k, m \in Fresh$, and let $b \in Role$. Consider the following set $\mathcal{T}$:*

$$\mathcal{T} = \{\{\!|\, k \,|\!\}_{\mathsf{pk}(b)}, \{\!|\, \mathsf{h}(m) \,|\!\}_k, \mathsf{sk}(b)\}.$$

It holds that $\mathcal{T} \vdash k$, since we have $\mathcal{T} \vdash \{\!|\, k \,|\!\}_{\mathsf{pk}(b)}$ and $\mathcal{T} \vdash \mathsf{sk}(b)$. It also holds that $\mathcal{T} \vdash \mathsf{h}(m)$, since we have $\mathcal{T} \vdash k$ and $\mathcal{T} \vdash \{\!|\, \mathsf{h}(m) \,|\!\}_k$. Moreover, we have $\mathcal{T} \vdash \mathsf{h}(k)$, since we have $\mathcal{T} \vdash k$. However, we have $\mathcal{T} \nvdash m$, since m cannot be derived from $\mathsf{h}(m)$.

2.3.2 Roles and Protocol Specifications

A security protocol consists of a collection of *roles*. A role describes the abstract sequences of *role events* that an agent might perform when executing his part of the protocol. We mainly focus on three types of events: send, recv and claim. The events recv and send concern the exchange of messages, while claim events are internal to an agent. To define claim events, we assume a set *Claim* of *claim events*, of which we will define the meaning below.

Definition 2.15 (Role events)**.** *We define the set RoleEvent of all role events as follows.*

$$\begin{aligned} \mathit{RoleEvent} ::= {} & \mathsf{send}(\mathit{Role}, \mathit{Role}, \mathit{RoleTerm}) \\ \mid {} & \mathsf{recv}(\mathit{Role}, \mathit{Role}, \mathit{RoleTerm}) \\ \mid {} & \mathsf{claim}(\mathit{Role}, \mathit{Claim}) \end{aligned}$$

The three events have the following meaning:

- $\mathsf{send}(R, R', t)$ denotes the sending of message t by role R with intended receiver R'.
- $\mathsf{recv}(R, R', t)$ denotes the reception of message t by role R', apparently sent by R.
- $\mathsf{claim}(R, p)$ denotes that role R claims security property p. Examples of such security properties will be introduced later. The idea behind claim events is that the security protocol assures that the property holds if an agent reaches such a claim.

Now we define the *active role* of a role event.

Definition 2.16 (Active role)**.** *If $e = \mathsf{send}(R, R', t)$, $e = \mathsf{recv}(R', R, t)$, or $e = \mathsf{claim}(R, p)$, we call R the* active role *of event e, denoted $\mathit{active}(e)$.*

Now that we have defined role events, we can specify how these role events can be combined to form *role specifications*. We define role specifications recursively. To start the recursion, we say that role events are the most basic form of role specifications. We consider five ways of composing role specifications. *Sequential composition* of roles R and S, denoted by $R \cdot S$, indicates that first R is executed, followed by S. *Alternative composition* of roles R and S, denoted by $R + S$, indicates that either R or S is executed. *Conditional branching* of roles R and S given role terms t and t', denoted by $(R \lhd t = t' \rhd S)$, indicates that R is executed if t equals t', and S otherwise. *No-exit iteration* [Fok96] of roles R, denoted by R^ω, indicates that R is executed infinitely often. *Binary Kleene composition* [Fok97] of role specifications R and S, denoted by $R * S$, indicates that R is executed zero or more times, followed by the execution of S.

Definition 2.17 (Role specifications)**.**

$$
\begin{aligned}
\mathit{RoleSpec} ::= {} & \mathit{RoleEvent} \\
| \ & \mathit{RoleSpec} \cdot \mathit{RoleSpec} \\
| \ & \mathit{RoleSpec} + \mathit{RoleSpec} \\
| \ & (\mathit{RoleSpec} \lhd \mathit{RoleTerm} = \mathit{RoleTerm} \rhd \mathit{RoleSpec}) \\
| \ & \mathit{RoleSpec}^\omega \\
| \ & \mathit{RoleSpec} * \mathit{RoleSpec}.
\end{aligned}
$$

Now we can define a *protocol specification* as a partial mapping from roles to role specifications.

Definition 2.18 (Protocol specification)**.** *A protocol specification is a partial function from Role to RoleSpec. We write Protocol for the set of all protocol specifications. We assume that Role is a fresh set of variables for every protocol.*

We only consider security protocols where:

- the specified role is the active role in all of its events, and
- all variables occurring in send or claim events have first been instantiated in a recv event.

Example 2.3. *We demonstrate the notions introduced so far on the following protocol.*

Let $S, R, T \in \mathit{Role}$*, let* $m, \mathsf{continue}, \mathsf{true}, \mathsf{false} \in \mathit{Const}$*, let* $V, W, X \in \mathit{Var}$ *and let* $\mathsf{NRO} = \{\!|\, m \,|\!\}_{\mathsf{sk}(S)}$ *and* $\mathsf{NRR} = \{\!|\, m \,|\!\}_{\mathsf{sk}(R)}$*.*

Protocol 2.1:

$$
\begin{aligned}
S \mapsto {} & \mathsf{recv}(T, S, \mathsf{continue}) \cdot \mathsf{send}(S, R, \mathsf{NRO}) \cdot \\
& (\mathsf{send}(S, T, \mathsf{false}) \ * (\mathsf{send}(S, T, \mathsf{true})) \cdot \mathsf{recv}(T, S, V))) \\
R \mapsto {} & \mathsf{send}(R, T, \mathsf{NRR}) \cdot \mathsf{recv}(S, R, V) \\
T \mapsto {} & \mathsf{recv}(R, T, V) \cdot \mathsf{send}(T, S, \mathsf{continue}) \cdot \\
& (\mathsf{recv}(S, T, \mathsf{false}) \ * (\mathsf{recv}(S, T, \mathsf{true}) \cdot \mathsf{send}(T, S, \mathsf{NRR})))
\end{aligned}
$$

We use an informal notation, called Alice-and-Bob notation*, to graphically depict security protocols in a way that is easier to read. Protocol 2.1 can be displayed in Alice-and-Bob notation as follows. In such a description, $R \to T : m$ means that R sends message m to T.*

Protocol 2.1

1. $R \to T$: NRR
2. $T \to S$: `continue`
3. $S \to R$: NRO
4. Choice for S:
   (a) i. $S \to T$: false
   ii. Go to 4.
   (b) i. $S \to T$: true
   ii. $T \to S$: NRR

In this protocol, first R sends NRR to T. Next, T sends `continue` to R. Them, S sends NRO to R. After that, S can send false to S zero or more times, followed by sending true to T. Finally, T sends NRR to S.

To each role we assign the *initial knowledge* that an agent is assumed to have when executing this role. We will normally not make this knowledge explicit when it can be derived from the context. For instance, if a role contains the signing of a message, we assume that the executing agent knows the signing key.

2.3.3 Runs

In the previous section, we have modeled a protocol as a set of roles. In this section, we will model the execution of a protocol. To do so, we assume a set $\mathbb{A}\mathrm{gt}$ of agents. An agent can execute more than one instance of the same role, or even several instantiations of different roles, at the same time.

We assume a set *RID* of *run identifiers*, and a set *AttackerFresh* of nonces generated by the attacker.

Now we will define the set of *run terms* which are used to build runs. The set of run terms is similar to the set of role terms, with the following differences. First, run terms can contain agents. Second, variables, nonces, and roles are marked with the run identifier, to avoid confusion between variables, nonces, or roles from different runs. Third, we add the set *AttackerFresh* of nonces generated by the attacker.

Definition 2.19 (Run Terms)**.** *The set of run terms RunTerms is defined as*

follows.

$$\begin{aligned} \mathit{RunTerms} ::= \quad & \mathit{Var}^{\#\mathit{RID}} \mid \mathit{Const} \mid \mathit{Fresh}^{\#\mathit{RID}} \mid \mathit{Role}^{\#\mathit{RID}} \mid \mathbb{A}\mathrm{gt} \\ \mid \ & \mathit{AttackerFresh} \\ \mid \ & (\mathit{RunTerms}, \mathit{RunTerms}) \\ \mid \ & \mathsf{h}(\mathit{RunTerms}) \\ \mid \ & \{\!|\, \mathit{RunTerms} \,|\!\}_{\mathit{RunTerms}} \\ \mid \ & \mathsf{sk}(\mathit{Role}) \mid \mathsf{pk}(\mathit{Role}) \mid \mathsf{k}((, \mathit{Role}), \mathit{Role}). \end{aligned}$$

We define the inverse of run terms analogous to the inverse of role terms.

Definition 2.20 (Inverse)**.** *We define the inverse function* $\ldots^{-1} : \mathit{RunTerms} \to \mathit{RunTerms}$ *as follows. Given* $R \in \mathit{Role} \cup \mathbb{A}\mathrm{gt}$*, we set* $\mathsf{pk}(R)^{-1} = \mathsf{sk}(R)$ *and* $\mathsf{sk}(R)^{-1} = \mathsf{pk}(R)$*. Moreover, we set* $t^{-1} = t$ *for all terms* t *that are not of the form* $\mathsf{sk}(R)$ *or* $\mathsf{pk}(R)$*.*

An instantiation is determined by three components:

- A *run identifier* θ. This is to bind together the different instantiated events that make up a run. Thus, it distinguishes different executions of the same role.
- A *role assignment* ρ. This is a function that maps the roles occurring in an event to actual agents executing these roles.
- A *variable assignment* σ. This is a function that maps variables to concrete values.

Definition 2.21 (Instantiation)**.** *An instantiation is a triple* (θ, ρ, σ)*, where* $\theta \in \mathit{RID}$ *is a run identifier,* $\rho \in \mathit{Role} \nrightarrow \mathbb{A}\mathrm{gt}$ *is a role assignment, and* $\sigma \in \mathit{Var} \nrightarrow \mathit{RunTerms}$ *is a variable assignment. We write Inst for the set of all instantiations.*

The *instantiation of event* e is denoted by $((\theta, \rho, \sigma), e)$.

Definition 2.22 (Instantiation of event)**.** *Let* $\mathit{inst} \in \mathit{Inst}$ *be an instantiation such that* $\mathit{inst} = (\theta, \rho, \sigma)$*. Let* rt, rt_1, rt_2 *be role terms such that* $\mathit{roles}(rt) \subseteq \mathit{dom}(\rho)$ *and* $\mathit{vars}(rt) \subseteq \mathit{dom}(\sigma)$*.*

$$(\mathit{inst}, rt) = \begin{cases} n^{\#\theta} & \text{if } rt = n \in \mathit{Fresh}, \\ \rho(R) & \text{if } rt = R \in \mathit{Role}, \\ \sigma(v) & \text{if } rt = v \in \mathit{Var}, \\ c & \text{if } rt = c \in \mathit{Const}, \\ ((\mathit{inst}, rt_1), (\mathit{inst}, rt_2)) & \text{if } rt = (rt_1, rt_2), \\ \mathsf{h}((\mathit{inst}, rt_1)) & \text{if } rt = \mathsf{h}(rt_1), \\ \{\!|\, (\mathit{inst}, rt_1) \,|\!\}_{(\mathit{inst}, rt_2)} & \text{if } rt = \{\!|\, rt_1 \,|\!\}_{rt_2}, \\ \mathsf{sk}((\mathit{inst}, rt_1)) & \text{if } rt = \mathsf{sk}(rt_1), \\ \mathsf{pk}((\mathit{inst}, rt_1)) & \text{if } rt = \mathsf{pk}(rt_1), \\ \mathsf{k}((\mathit{inst}, rt_1), (\mathit{inst}, rt_2)) & \text{if } rt = \mathsf{k}(rt_1, rt_2). \end{cases}$$

Definition 2.23 (Set of all runs)**.** *We define the set Run of all runs as follows.*

$$Run = Inst \times RoleEvent^*,$$

where $RoleEvent^*$ denoted a sequence of elements of type RoleEvent.

A run is an instantiation of a role, and consists of the instantiations of the role's events. To be more precise, we denote the instantiation of a sequence $e_1, \ldots, e_n$ of events by $((\theta, \rho, \sigma), (e_1, \ldots, e_n))$. An instantiation of a sequence of events is equal to a sequence of instantiated events:

$$((\theta, \rho, \sigma), (e_1, \ldots, e_n)) = ((\theta, \rho, \sigma), e_1), \ldots, ((\theta, \rho, \sigma), e_n).$$

The *actor* $actor((inst, e))$ of an instantiated event $(inst, e)$ is the agent who executed the active role.

Definition 2.24 (Actor)**.** *The actor of an instantiated event is determined by a function actor from instantiated events to agents that is defined as follows.*

$$actor((\theta, \rho, \sigma), e) = \rho(active(e)).$$

If $s = P(R)$, then we define the set of possible runs of this role by

$$runsof(P, R) = \{((\theta, \rho, \emptyset), s) \mid \theta \text{ a run identifier}, \rho \text{ maps all roles in } P \text{ to agents}\}.$$

Definition 2.25 (Run Event)**.** *The set RunEvent of run events is defined as follows.*

$$Inst \times (RoleEvent \cup \{\mathsf{create}(R) \mid R \in Role\}).$$

Furthermore, we define the set of *active run identifiers* given a set of runs F.

Definition 2.26 (Active Run Identifiers)**.** *Given a set of runs F, we define the set of active run identifiers as*

$$runIDs(F) = \{\theta \mid ((\theta, \rho, \sigma), s) \in F\}.$$

2.3.4 Protocols as Concurrent Game Structures

Now we will define an operational semantics for security protocols, by representing security protocols as infinite concurrent game structures.

We have seen that an infinite concurrent game structure consists of six components, i.e., $M = (\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta)$. We will now define these components individually.

We use an infinite set of agents, denoted $\mathbb{A}\mathrm{gt}$, as agents of the concurrent game structure. One of the agents is a dedicated *scheduler Sched*, i.e., we assume that $Sched \in \mathbb{A}\mathrm{gt}$. The actions of the concurrent game structure are the set of run events, together with the empty move ϵ.

Definition 2.27 (Actions of protocols represented as a concurrent game structure)**.** *The set of actions of protocols that are represented as a concurrent game structure are defined as follows.*

$$Act = RunEvent \cup \{\epsilon\}.$$

The states of a concurrent game structure consist of three components. The first component represent the agent whose turn it is. The second component represents the attacker knowledge. The third component represents the (remainder of the) runs that still have to be executed.

Definition 2.28 (States of protocols represented as a concurrent game structure)**.** *The set of states of protocols that are represented as a concurrent game structure are defined as follows.*

$$Q = \mathbb{A}\mathrm{gt} \times \mathcal{P}(RunTerms) \times \mathcal{P}(Run).$$

The labeling function π and the set of propositions Π depend on the purpose of the verification. We therefore define these components in the chapters in which they are used.

We proceed by defining the transition relation. The transition relation will be defined in two steps, following Van Deursen [VD11]. The actual transition relation is defined by *agent rules*, which defines under which conditions an agent may execute an event. Agent rules are defined in terms of *composition rules*, which are used to model protocol flow.

Composition rules indicate transitions from one role specification to another by means of an event. They have the form $s_1 \overset{e}{\Rightarrow} s_2$, with $s_1, s_2 \in RoleSpec$ and $e \in RoleEvent$. We write $s_1 \overset{e}{\Rightarrow} s_2$ for transactions from role specification s_1 to role specification s_2 by means of event e. The composition rules are displayed in Table 2.1, for $s_1, s_1', s_2, s_2' \in Agenda$ and $e \in RoleEvent$.

The rule exec states that if we have atomic event e, we can execute that event and be done. Rules seq_1 and seq_2 states that in $s_1 \cdot s_2$, we can either completely execute s_1 and continue with s_2, or do a step in s_1 and continue with the rest of s_1, followed by s_2. The rules choice_L and choice_R states that it is possible to execute a step from either the left or the right branch of parallel composition, and to discard the other branch. The rules kleene_1, kleene_2, kleene_3, and kleene_4 state that in $s_1 * s_2$, we can completely execute s_1 and execute $s_1 * s_2$ again, or partially execute s_1 and execute the rest of s_1 followed by $s_1 * s_2$, or completely execute s_2 and be done, or partially execute s_2 and execute the rest of s_2. The rules inf_1 and inf_2 state that in ${s_1}^\infty$, we can either fully execute s_1 and execute s_1^∞ again, or partially execute s_1 and execute the rest of s_1, followed by s_1^∞ again.

Next, we define the set of all paths by defining *derivation rules*, which define when an agent can cause the system to transition from one state to another Before we can do so, we first need to define *matching*. During the execution of a protocol, variables are assigned values by executing recv events. The term in the recv event can be seen as a pattern that should be matched by the term in an incoming message. We write

$$[\mathsf{exec}]\frac{}{e \overset{e}{\Rightarrow} \checkmark}$$

$$[\mathsf{seq}_1]\frac{s_1 \overset{e}{\Rightarrow} \checkmark}{s_1 \cdot s_2 \overset{e}{\Rightarrow} s_2} \qquad [\mathsf{seq}_2]\frac{s_1 \overset{e}{\Rightarrow} s_1'}{s_1 \cdot s_2 \overset{e}{\Rightarrow} s_1' \cdot s_2}$$

$$[\mathsf{choice}_\mathsf{L}]\frac{s_1 \overset{e}{\Rightarrow} s_1'}{s_1 + s_2 \overset{e}{\Rightarrow} s_1'} \qquad [\mathsf{choice}_\mathsf{R}]\frac{s_2 \overset{e}{\Rightarrow} s_2'}{s_1 + s_2 \overset{e}{\Rightarrow} s_2'}$$

$$[\mathsf{cond}_\mathsf{T}]\frac{s_1 \overset{e}{\Rightarrow} s_1'}{s_1 \lhd x = x \rhd s_2 \overset{e}{\Rightarrow} s_1'} \qquad [\mathsf{cond}_\mathsf{F}]\frac{s_2 \overset{e}{\Rightarrow} s_2' \quad x \neq y}{s_1 \lhd x = y \rhd s_2 \overset{e}{\Rightarrow} s_2'}$$

$$[\mathsf{kleene}_1]\frac{s_1 \overset{e}{\Rightarrow} \checkmark}{s_1 * s_2 \overset{e}{\Rightarrow} s_1 * s_2} \qquad [\mathsf{kleene}_2]\frac{s_1 \overset{e}{\Rightarrow} s_1'}{(s_1 * s_2) \overset{e}{\Rightarrow} s_1' \cdot (s_1 * s_2)}$$

$$[\mathsf{kleene}_3]\frac{s_2 \overset{e}{\Rightarrow} \checkmark}{s_1 * s_2 \overset{e}{\Rightarrow} \checkmark} \qquad [\mathsf{kleene}_4]\frac{s_2 \overset{e}{\Rightarrow} s_2'}{s_1 * s_2 \overset{e}{\Rightarrow} s_2'}$$

$$[\mathsf{inf}_1]\frac{s_1 \overset{e}{\Rightarrow} \checkmark}{{s_1}^\infty \overset{e}{\Rightarrow} {s_1}^\infty} \qquad [\mathsf{inf}_2]\frac{s_1 \overset{e}{\Rightarrow} s_1'}{{s_1}^\infty \overset{e}{\Rightarrow} s_1' \cdot {s_1}^\infty}$$

Table 2.1: Composition rules

$$\mathrm{Match}((\theta, \rho, \sigma), pt, t, (\theta, \rho, \sigma'))$$

to denote the process of matching pattern pt by term t, where σ' is the extension of substitution σ by binding variables from pt to values from t.

Definition 2.29 (Match [CM03, CM12])**.** *For every instantiation $inst = (\theta, \rho, \sigma) \in Inst$, role term $pt \in RoleTerm$, run term $m \in RunTerms$, and instantiation $inst' = (\theta', \rho', \sigma') \in Inst$, the predicate* $\mathrm{Match}(inst, pt, m, inst')$ *holds if and only if $\theta = \theta'$, $\rho = \rho'$ and*

$$\begin{aligned} &(inst', pt) = m \wedge \\ &\sigma \subseteq \sigma' \wedge \\ &dom(\sigma') = dom(\sigma) \cup vars(pt). \end{aligned}$$

Example 2.4. *The following predicate is true:*

$$\mathrm{Match}((\theta, \rho, \emptyset), \mathsf{recv}(R, S, \{\!|\, V, n \,|\!\}_{\mathsf{pk}(R)}), ((\theta, \rho, \emptyset), \{\!|\, c, n \,|\!\}_{\mathsf{pk}(R)}), (\theta, \rho, \{V \mapsto c\})).$$

This predicate expresses that the instantiated event $((\theta, \rho, \emptyset), \mathsf{recv}(R, S, \{\!|\, V, n \,|\!\}_{\mathsf{pk}(R)}))$ matches a term of the form $((\theta, \rho, \emptyset), \{\!|\, c, n \,|\!\}_{\mathsf{pk}(R)})$. The result of this match is that the instantiation $(\theta, \rho, \emptyset)$ is extended to $(\theta, \rho, \{V \mapsto c\})$.

Now we can define the *derivation rules* (Table 2.2). The first rule in this table defines the creation of a new run for role R. Such a new run is assigned a fresh

$$[\mathsf{create}]\frac{P \in Protocol \quad r \in dom(P) \quad (inst, s) \in runsof(P, r) \quad \theta \notin runIDs(F) \quad inst = (\theta, \rho, \emptyset) \quad a \neq Sched}{(a, AKN, F) \xrightarrow{(inst, \mathsf{create}(r))}_a (S, AKN, F \cup \{(inst, s)\})}$$

$$[\mathsf{send}]\frac{(inst, s) \in F \quad (inst, s) \xrightarrow{(inst, e)} (inst, s') \quad e = \mathsf{send}(r, r', m) \quad a \neq Sched}{(a, AKN, F) \xrightarrow{(inst, e)}_a (Sched, AKN \cup \{(inst, m)\}, (F \setminus (inst, s)) \cup (inst, s'))}$$

$$[\mathsf{recv}]\frac{(inst, s) \in F \quad (inst, s) \xrightarrow{(inst', e)} (inst', s') \quad e = \mathsf{recv}(r, r', pt) \quad AKN \vdash m \quad \mathrm{Match}(inst, pt, m, inst') \quad a \neq Sched}{(a, AKN, F) \xrightarrow{(inst', e')}_a (S, AKN, (F \setminus (inst, s)) \cup (inst', s'))}$$

$$[\mathsf{claim}]\frac{(inst, s) \in F \quad (inst, s) \xrightarrow{(inst, e)} (inst, s') \quad e = \mathsf{claim}(r, p) \quad a \neq Sched}{(a, AKN, F) \xrightarrow{(inst, e)}_a (S, AKN, (F \setminus (inst, e \cdot s)) \cup (inst, s))}$$

$$[\mathsf{empty}]\frac{a \neq Sched}{(a, AKN, F) \xrightarrow{\mathsf{empty}}_a (S, AKN, F)}$$

$$[\mathsf{schedule}]\frac{a \in \mathbb{A}\mathrm{gt}}{(S, AKN, F) \xrightarrow{\mathsf{schedule}}_{Sched} (a, AKN, F)}$$

Table 2.2: Operational semantics rules.

run identifier ($\theta \notin runIDs(F)$). The second rule defines the execution of a send event. We model the attacker to have full control over the network. Thereto, it is in fact the attacker, who plays the role of the network, that receives the message from R, and who can forward the message to R'. The condition of the rule states that there is a run starting with a send event. The conclusion expresses that the attacker knowledge is extended with the sent message and that the executed send event is dropped from the run, allowing the run to proceed with the next event. The third rule defines the execution of a recv event. The condition states that the receiving pattern matches a term derivable from the attacker knowledge. This models the fact that the attacker provides input to the receiving agent. The state of the corresponding run is updated through the instantiation of the variables occurring in the pattern (from $inst$ to $inst'$). The fourth rule expresses that claim events are simply executed without significantly affecting the system state. The fifth rule is an empty event, which agents can use to give back the turn to the scheduler. The sixth rule states that the scheduler can give the turn to any event.

We define the transition relation in the concurrent game structure based on the derivation rules.

Definition 2.30 (Transition relation of protocols represented as a concurrent game structure)**.** *The transition relation of protocols that are represented as a concurrent*

game structure is the smallest function δ such that if $q \xrightarrow{\alpha}_{a_i} q'$ is a valid derivation rule, then we have $\delta(q, (\mathsf{nil}, \ldots, \mathsf{nil}, \alpha, \mathsf{nil}, \ldots) = q'$, where α occurs on position i in the tuple.

Finally, we define the move function as follows:

Definition 2.31 (Move function of protocols represented as a concurrent game structure)**.** *The move function for agent a_i of protocols that are represented as a concurrent game structure is the smallest function d_i such that if $q \xrightarrow{\alpha}_{a_i} q'$ is a valid derivation rule, then we have $\alpha \in d_i(q)$, and for each state $q = (a, AKN_0, F)$ and agent a_j such that $a_j \neq a_i$, we have $\mathsf{nil} \in d_j(q)$.*

The initial state of the system, denoted by s_0, is defined by $(Sched, AKN_0, \emptyset)$, where $\emptyset$ expresses that there are initially no runs. AKN_0 denotes the initial attacker knowledge. This knowledge includes all publicly known data. Further, it contains the initial knowledge of all dishonest agents. The empty set $\emptyset$ is used here to denote that none of the variables has a value (yet).

Now we define *executions* and *traces* given this transition system.

Definition 2.32 (Execution, trace)**.** *An execution is a (possibly infinite) sequence of the form $s_0, t_1, s_1, t_2, s_2, \ldots$, where s_i are states of the form (a, AKN, F), and t_i are instantiated events of the form $(inst, e)$ or $(inst, \mathsf{create}(r))$. Every step in the execution must correspond to a transition $s_i \xrightarrow{t_{i+1}}_a s_{i+1}$ which is derivable from the above transition system. The (possibly infinite) sequence $t_1, t_2, t_3, \ldots$ is called a trace of the protocol. Given a trace t, we write t_i for the i'th element of t.*

2.3.5 Resilience

Some security properties require the existence of a future event. For example, every time an agent in a non-repudiation protocol has received evidence of an event, he should be sure that eventually, J will accept this evidence. Such properties are also called *liveness properties*. Liveness properties, in general, do not hold in the Dolev-Yao attacker model. Therefore, we assume an attacker that has less power over the communication channels than the standard *Dolev-Yao attacker*. To be more precise, for the liveness properties, we assume *resilient channels* [Aso98, CD06], which are communication channels that eventually deliver all transmitted messages. We will formalize resilient channels by assuming *resilient agents*. For simplicity of modeling, we follow [CD06] by assuming that if a message is submitted multiple times, it only needs to be delivered once. We defend this assumption by pointing out that in typical good protocols, messages are never delivered multiple times. Note that if in practice no resilient channels exist from agent a to agent b, it can be simulated by a *trusted third party* T that relays messages from a to b, whenever the channels between a and T and between T and b are resilient. We say that an agent is resilient if the agent does not postpone enabled actions infinitely often.

More precisely, we say that an agent is resilient if he does not indefinitely postpone making claims, sending messages, or receiving pending messages, if specified so by his role description (unless the execution of such an event is forever blocked by

the other agents). Also, we require that an agent does not indefinitely postpone blocking create events. We implement this by requiring that the agent executes such instantiated events fairly, i.e., these events must not be enabled at the end of the trace (for finite traces), or must occur infinitely often if they are enabled infinitely often (for infinite traces). Note that the notion of fairness discussed here is not related to fairness in the sense of fair exchange.

We proceed to define *resilience* in security protocols. To do so, we first define the *protocol corresponding to instantiated event*, *enabled* instantiated events, *pending* recv, and *blocking* create events.

Definition 2.33 (Protocol corresponding to instantiated event)**.** *The protocol corresponding to an instantiated event $t_i = ((\theta, \rho, \sigma), e)$ in trace t, written $protocol_t(t_i)$, is the protocol P for which there exists $t_j = ((\theta, \rho', \sigma'), \mathsf{create}(R))$ such that $R \in dom(P)$.*

Definition 2.34 (Enabled)**.** *We write $en_t(P)$ for the set of* enabled *instantiated events of protocol P in trace t, i.e., those events s such that $t' = t_1, \ldots, t_n, s$ is also a trace and $protocol_{t'}(s) = P$.*

A *pending* recv event is a recv event of a term that has been sent but not yet received.

Definition 2.35 (Pending recv)**.** *Instantiated event $e = ((\theta, \rho, \sigma), \mathsf{recv}(R_1, R_2, t))$ is a pending* recv *event in finite trace $t = t_1, \ldots, t_i$ if there exists $j \leq i$ and instantiation $(\theta', \rho', \sigma')$ such that $t_j = ((\theta', \rho', \sigma'), \mathsf{send}(R_1', R_2', t'))$ with $\rho(R_2) = \rho'(R_2')$ and $\sigma(t) = \sigma'(t')$, and there does not exist k $(j < k \leq i)$ and instantiation $(\theta'', \rho'', \sigma''$ such that $t_k = ((\theta'', \rho'', \sigma''), \mathsf{recv}(R_1'', R_2'', t''))$ with $\rho(R_2) = \rho''(R_2'')$ and $\sigma(t) = \sigma''(t'')$.*

A *blocking* create event is a create event that must be executed in order to execute a claim event or a pending receive event.

Definition 2.36 (Blocking create)**.** *An instantiated event $e = \mathsf{create}(r)$ is a blocking* create *event in trace t if $r = e_1, \ldots$ and e_1 is a* claim *or pending* recv *event such that $e_1 \notin en_t(P)$ and $e_1 \in en_{t,(inst,e)}(P)$ for some instantiation $inst$.*

Now we can formally define resilience of an agent.

Definition 2.37 (Resilient)**.** *Agent a is* resilient *in executing protocol P in trace t, if a executes P* fairly*, i.e., if $s = (inst, e)$ such that $actor(s) = a$ and s is either a pending* recv *event, a* send *event, a* claim *event, or a blocking* create *event in t, then:*

- *$s \notin en_t(P)$ (if t is finite);*
- *if $s \in en_{t_1,\ldots,t_j}(P)$ for infinitely many j, then for infinitely many j', $t_{j'} = (inst_j', e)$ where $inst_j'$ is an instantiation (if t is infinite).*

2.4 Alternating-time Temporal Logic

Security properties are properties of protocols. Since we model protocols as concurrent game structures. we model security properties as properties of concurrent game structures. To do so, we use ATL*, a version of *Alternating-time Temporal Logic*. ATL* can be seen as an extension of *Computation Tree Logic* (CTL*) [EH85]. For those readers familiar with CTL*, the difference is that in ATL*, the quantification over paths is replaced by quantification over strategies. The logic is particularly suited for reasoning about strategic properties, as it allows to express that there exists a *strategy* with which an agent obtains a desired property, instead of requiring that all protocol runs have to satisfy the property, independent of the agent's behavior.

ATL* makes use of *path formulas*, i.e., formulas that are properties of a single path in the concurrent game structure. Path formulas correspond to LTL formulas.

We proceed by defining ATL*, one of the variants of *Alternating-time Temporal Logic* [AHK02]. ATL* has a *strategic operator* $\langle\langle C \rangle\rangle$, which can be seen as a path quantifier that ranges over all paths into which the agents in C can force the game.

Path formulas can contain the temporal operators $\bigcirc$ ('next'), $\Box$ ('always'), and $\mathcal{U}$ ('until'). These operators can be combined. For example, $\bigcirc\Box\varphi$ means that from the next state on, φ will always hold.

We proceed by defining the set of ATL* *state formulas* and ATL* *path formulas*.

Definition 2.38 (ATL* state formulas, ATL* path formulas [AHK02])**.** *We assume a set of propositions* Π *and a set of agents* $\mathbb{A}\mathrm{gt}$*. We define the set of* ATL* *state formulas* Φ *and the set of* ATL* *path formulas* Ψ *as follows.*

$$\Phi ::= \Pi \mid \neg\Phi \mid \Phi \wedge \Phi \mid \langle\langle \mathcal{P}(\mathbb{A}\mathrm{gt}) \rangle\rangle \Psi;$$
$$\Psi ::= \Phi \mid \neg\Psi \mid \Psi \vee \Psi \mid \bigcirc\Psi \mid \Box\Psi \mid \Psi \,\mathcal{U}\, \Psi.$$

We call ATL* *state formula sometimes simply* ATL* *formulas.*

ATL* state formulas are defined in a state in a concurrent game structure. ATL* path formulas are defined in a path in a concurrent game structure.

Definition 2.39 (Interpretation of ATL*)**.** *We write* $M, q \models \varphi$ *for the interpretation of* ATL* *state formula* φ *in state* $q \in Q$ *of concurrent game structure* $M = (\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta)$*. We write* $M, q \not\models \varphi$ *when it does not hold that* $M, q \models \varphi$*.*

The interpretation of ATL* *state formulas is defined as follows.*

- $M, q \models p$ *for propositions* $p \in \Pi$*, if* $p \in \pi(q)$*.*
- $M, q \models \neg\varphi$ *if* $M, q \not\models \varphi$*.*
- $M, q \models \varphi_1 \wedge \varphi_2$ *if* $M, q \models \varphi_1$ *and* $M, q \models \varphi_2$*.*
- $M, q \models \langle\langle C \rangle\rangle\varphi$ *if there exists a collective strategy* σ_C *such that for all paths* $\lambda \in out(q, \sigma_C)$ *and positions* $i \geq 0$*, we have* $M, \lambda[i, \infty] \models \varphi$*.*

We write $M, \lambda \models \varphi$ *for the interpretation of* ATL* *path formula* φ *in path* $\lambda \in Q^*$ *of concurrent game structure* $M = (\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta)$. *We write* $M, \lambda \not\models \varphi$ *when it does not hold that* $M, \lambda \models \varphi$.

The interpretation of ATL* *path formulas is defined as follows.*

- $M, \lambda \models \top$.
- $M, \lambda \not\models \bot$.
- $M, \lambda \models \varphi$ *where* $\varphi \in \Phi$ *if* $M, \lambda[0] \models \varphi$.
- $M, \lambda \models \neg\varphi$ *if* $M, \lambda \not\models \varphi$.
- $M, \lambda \models \varphi_1 \vee \varphi_2$ *if* $M, \lambda \models \varphi_1$ *or* $M, \lambda \models \varphi_2$.
- $M, \lambda \models \bigcirc\varphi$ *if* $M, \lambda[1, \infty] \models \varphi$.
- $M, \lambda \models \Box\varphi$ *if for all* $i \geq 0$, *we have* $M, \lambda[i, \infty] \models \varphi$.
- $M, \lambda \models \varphi_1 \,\mathcal{U}\, \varphi_2$ *if there exists* $i \geq 0$ *such that* $M, \lambda[i, \infty] \models \varphi_2$ *and for all* $0 \leq j < i$ *we have* $M, \lambda[j, \infty] \models \varphi_1$.

We write $M, \lambda \models \Diamond\varphi$ *for* $M, \lambda \models \neg\Box\neg\varphi$. *Furthermore, we write* $\models \varphi$ *if* $M, q \models \varphi$ *for all concurrent game structures* M *and all states* q *of* M.

The universal path quantifier of the branching-time temporal logic CTL can be captured in ATL as $\forall \equiv \langle\langle\emptyset\rangle\rangle$. The existential path quantifier $\exists$ will be interpreted as usual in CTL.

The following example illustrates the interpretation of ATL*.

Example 2.5. *Consider concurrent game structure* M *from Figure 2.1. We have* $M, q_0 \models \langle\langle a\rangle\rangle\Diamond p_a$ *and* $M, q_0 \models \langle\langle a, b\rangle\rangle(\Diamond p_a \wedge \Diamond p_b)$, *but* $M, q_0 \not\models \langle\langle a\rangle\rangle\Diamond p_c$.

2.5 Conclusion

We provided a game-based operational semantics for security protocols. We did so by unifying the Cremers-Mauw semantics with concurrent game structures. This allows us to define security properties on Cremers-Mauw protocol specifications in Alternating-time Temporal Logic. In the next chapters, we will use this semantics for the study of non-repudiation and fair-exchange protocols.

3

Imperfect Information in Fair Non-repudiation Protocols

Abstract. We indicate two problems with the specifications of fairness that are currently used for the verification of non-repudiation and other fair-exchange protocols. The first of these problems is the implicit assumption of perfect information. The second problem is the possible lack of effectiveness. We solve both problems in isolation by giving new definitions of fairness, but leave the combined solution for the next chapter. Moreover, we establish a hierarchy of various definitions of fairness, and indicate the consequences for existing work.

3.1 Introduction

The correctness of a security protocol depends in general on the precise formulation of its security properties. Consequently, the development of appropriate security properties is at least as important as the proper design of security protocols. Classical properties, such as *confidentiality* and *authentication*, are well understood and have been exhaustively investigated [Ros96, Low97, CMdV06]. Research on more recent properties, such as *receipt-freeness* in electronic voting protocols [BT94, DKR06], seems to converge, while for other properties, such as *ownership transfer* in RFID protocols, discussions have only recently started [vDMRV09].

In this chapter, we study the development of the property of *fairness* for *non-repudiation protocols*. Recall that the main goal of a non-repudiation protocol is to allow two (or more) parties to exchange goods or messages without any of the parties being able to falsely deny having taken part in the exchange. Such a protocol is designed so that the sender of the message obtains a *non-repudiation of receipt* (NRR) evidence and the receiver of the message a *non-repudiation of origin* (NRO) evidence. An important property of fair-exchange protocols is *fairness*, which roughly states that if the receiver obtains NRO, then the sender can obtain NRR, and vice versa. An example of a non-repudiation protocol is a *certified e-mail* protocol [ASW98].

Although other properties, such as *abuse-freeness*, also apply to non-repudiation protocols (and the wider class of *fair-exchange protocols*), we will only investigate fairness and its relation to *effectiveness* and *strategic timeliness*. Effectiveness (sometimes also called *viability*) is not a security property, but a functional property, stating that the protocol can actually achieve the exchange of an NRR and an NRO evidence. Strategic timeliness requires that an agent always has an honest

strategy to stop execution of the protocol.

In the literature on non-repudiation protocols, a variety of different interpretations of the fairness property have been described. Most of these were formalized in the modal logic ATL* [AHK02] as to allow for the automated verification of protocols through model checking, for example in the Mocha model checker [AHM+98]. The observed variations seem to be due to differences in the assumed execution models of the agents involved, to differences in the attacker model, and to differences in the intended application of the protocol. Some authors already provided insight in the relation between some of the fairness definitions [CKS06].

Nevertheless, we observe two limitations of the existing definitions. The first concerns the implicit assumption of *perfect information*, as it is called in game theory. By this we mean that, at each moment, all agents have full knowledge of the global state of the system. In practice this does not seem a realistic assumption for a security protocol. One would expect an agent to only know his own state and use a protocol to infer knowledge of the other agents' states. This assumption has a significant impact on the formulation of fairness in ATL*.

The second limitation concerns the combination of fairness and effectiveness. In the game-theoretical setting, both properties are expressed in terms of the existence of strategies. By taking the conjunction of the two properties, one does not necessarily obtain a single strategy that enforces both fairness and effectiveness. Here, we propose a new property which blends fairness and effectiveness properly.

This chapter is structured as follows. We start with an example fair-exchange protocol in Section 3.2. In Section 3.3, we revisit existing notions of fairness. We introduce a notion of fairness based on the assumption of imperfect information in Section 3.4. In Section 3.5, we combine fairness and effectiveness. In Section 3.6, we develop the hierarchy of fairness properties and prove correctness and strictness of the inclusions. Finally, we consider implications for the practical use of various notions of fairness in the literature in Section 3.7, and conclude in Section 3.8.

3.2 Fair-exchange Protocols

Protocol 3.1

1. $a \to b$: m, NRO
2. $b \to a$: NRR

Protocol 3.1 is an example of a simple non-repudiation protocol, where Alice and Bob exchange non-repudiation of origin and receipt of message m. The protocol specifies that first Alice sends message m and NRO to Bob, and then Bob sends NRR to Alice. Here, NRO could be implemented as $[f_{\mathsf{NRO}}, a, b, m]_a$ and NRR as $[f_{\mathsf{NRR}}, a, b, m]_b$, and f_{NRR} and f_{NRO} are so-called *flags* indicating the type of the evidence. Note that this protocol is not fair, as Bob can abort after step 1, leaving Alice without NRR. Protocol 3.2 is an example of a fair NR-protocol (with inline TTP). Fairness is intuitively guaranteed because the TTP will not send out NRO and NRR before he has collected both evidences.

Protocol 3.2

1. $a \to t$: m, NRO
2. $t \to b$: m
3. $b \to t$: NRR
4. $t \to b$: NRO
5. $t \to a$: NRR

Non-repudiation protocols with inline TTP are generally inefficient, as the TTP becomes easily a bottleneck. Protocols with offline TTP do not suffer from this problem, but also tend to be more complex, as they typically contain non-determinism and various sub-protocols. This means that it is less easy to check by hand that fairness is satisfied. Therefore, a formal way of verifying fairness is needed.

In this chapter, we focus on fair exchange, and are therefore only interested in attacks carried out by the protocol participants themselves. We do not consider attacks carried caused by lack of secrecy or authentication, for example. We therefore assume *secure communication channels*. We implement this by assuming that every message m is signed, encrypted, and labeled with the name of the intended recipient. In other words, when we write $A \to B : m$, we mean in fact $A \to B : \{\!|\, \{\!|\, (m, B) \,|\!\}_{\mathsf{sk}(A)} \,|\!\}_{\mathsf{pk}(B)}$. Moreover, for reasons of clarity, we leave the scheduler (and communication channels) out of the specifications.

3.3 Existing Formalizations

Various definitions of fairness in ATL* have been proposed in the literature on non-repudiation protocols and other fair-exchange protocols. In this section, we first give an overview of the proposed definitions.

We assume that Alice sends a message to Bob. Alice wants to receive NRO of this message, and Bob wants to receive NRR of this message. One can distinguish *fairness for Alice* (whenever Bob receives NRO, Alice is guaranteed to obtain NRR), and *fairness for Bob* (whenever Alice obtains NRR, Bob is guaranteed to obtain NRO). We only consider fairness for Alice; fairness for Bob can be formulated symmetrically.

Fairness for an agent only needs to be guaranteed when the agent complies with the protocol: if an agent does not follow the protocol, he does so at his own risk. An agent that complies with the protocol is called *honest*. Fairness should be guaranteed for honest agents, even if the other agents are *dishonest*, i.e., behave in a way that is not foreseen by the protocol. Therefore, when studying fairness for Alice, we assume that Alice is honest, and that Bob might be dishonest. We do not require that fairness can be guaranteed after unintended dishonest behavior caused by system failures, as has been considered in [ES03, LNJ01].

We write $M, q \models \mathsf{NRO}$ whenever Bob has received NRO, and $M, w \models \mathsf{NRR}$ when-

ever Alice has received NRR. We assume that Alice cannot distinguish between states where Bob has or has not received NRO, everything else being equal, and that Bob cannot distinguish between states where Alice has or has not received NRR, everything else being equal.

Strong Fairness One of the definitions of fairness that are proposed by Kremer and Raskin [KR03] is *strong fairness*. It can be formulated as

$$\textsc{StrongFair} \equiv \forall\Box(\mathsf{NRO} \rightarrow \forall\Diamond\mathsf{NRR}).$$

Strong fairness for Alice states that in every reachable state where Bob has NRO, Alice should eventually obtain NRR, whatever the agents do. Strong fairness can be seen as *enforced fairness*: if the protocol is non-deterministic due to underspecification and thus gives Alice multiple available strategies, each of these strategies should guarantee her NRR.

Non-enforced Fairness If we assume that Alice is rational, StrongFair is stronger than necessary. A weaker form of fairness, which requires Alice to play rational, has also been proposed by Kremer and Raskin [KR03]. This form is called *non-enforced fairness*, and defined as

$$\textsc{NEFair} \equiv \neg\langle\!\langle b\rangle\!\rangle\Diamond(\mathsf{NRO} \wedge \neg\langle\!\langle a_h\rangle\!\rangle\Diamond\mathsf{NRR}).$$

This formula states that Bob should not have a strategy to reach a state where he has obtained NRO, while Alice at the same time does not have a strategy to obtain NRR. It is called *non-enforced fairness*, because a protocol that satisfies this property does not enforce fairness: it is possible that Alice can choose to play a strategy with which she might end in an unfair situation, as long as there exists at least one "good" strategy that guarantees Alice a fair situation.

Strategic Fairness Chadha et al. [CKS06] have proposed a notion of fairness in between strong fairness and non-enforced fairness, called *strategic fairness*, which is defined as

$$\textsc{StratFair} \equiv \forall\Box(\mathsf{NRO} \rightarrow \langle\!\langle a_h\rangle\!\rangle\Diamond\mathsf{NRR}).$$

A protocol satisfies *strategic fairness* for Alice if and only if in every reachable state, it holds that whenever Bob has obtained NRO, there exists a strategy for honest Alice that gives her NRR.

This definition, however, seems counterintuitive, as it combines the enforced and non-enforced approach. If one assumes that Alice is rational enough to resolve non-determinism in the correct way, then it is not necessary to require that she obtains the fair situation $\mathsf{NRO} \rightarrow \langle\!\langle a_h\rangle\!\rangle\Diamond\mathsf{NRR}$ independently of her strategy; it would suffice if there *exists* a strategy for Alice that guarantees the fair situation. On the other hand, if one does not assume that Alice is able to resolve non-determinism in the correct way, then it is not enough to require that there *exists* a strategy that gives her NRR; she might still never obtain NRR when she never plays the right strategy. Therefore, strategic fairness seems too strong for rational agents, and too weak for agents that are not rational.

Weak Fairness Another definition of fairness, proposed by Chadha et al. [CKS06], is *weak fairness*. This notion has been proposed for technical purposes to simplify verification. It is defined by

$$\textsc{WeakFair} \equiv \forall\Box(\mathsf{NRO} \rightarrow \exists\Diamond\mathsf{NRR}).$$

A protocol satisfies weak fairness for Alice if and only if in every reachable state, it holds that whenever Bob has obtained NRO, if all agents cooperate, Alice will eventually get NRR.

Invariant Fairness One disadvantage with the above formulations of fairness is that it is not sufficient to look at a single path to prove that a formula does not hold. Instead, to find a counterexample, the entire model needs to be taken into account. An alternative definition of fairness is proposed based on invariants. *Invariant fairness* [CKS06] for Alice only tests those states in which Alice has stopped the protocol, allowing counterexamples to be expressed as paths. The proposition Stop_a is defined to be true exactly when Alice has stopped executing the protocol. It is assumed that as soon as Alice has stopped executing the protocol, she cannot obtain NRR anymore, i.e., $\forall\Box((\text{Stop}_a \wedge \neg\mathsf{NRR}) \rightarrow \forall\Box\neg\mathsf{NRR})$. Now invariant fairness is defined by

$$\textsc{InvFair} \equiv \forall\Box(\text{Stop}_a \rightarrow (\mathsf{NRO} \rightarrow \mathsf{NRR})).$$

This formula states that in all states where Alice has stopped executing the protocol, Alice should possess NRR whenever Bob possesses NRO.

3.4 Fair Exchange and Imperfect Information

When ATL* formulas are interpreted in concurrent game structures, it is implicitly assumed that agents know precisely the current global state of the system, including the local states of the other agents [AHK02]. In other words, ATL* formulas are usually evaluated in a model that assumes *perfect information*. This is also the way in which the Mocha model checker [AHM+98] evaluates ATL* formulas. The first limitation of the current way of modeling fairness in ATL* is that the perfect-information assumption is unrealistic for communication protocols: if all agents knew the local state of all other agents, no communication would be needed. We will demonstrate that assuming perfect information leads to problems in practice, by showing that evaluating NEFair in imperfect-information models, as is done in [KR03], leads to counterintuitive results.

The problem is, basically, that an agent's perfect-information strategy can be *non-executable* under imperfect information: the strategy might require executing different actions in situations that look the same to the agent. Furthermore, even if an agent has an executable strategy, he may be unaware of having it, and unable to identify it [JvdH04]. For example, one can construct a protocol in which the message that Alice needs to send depends on whether Bob did or did not receive some other message. Alice does not know which messages have been received by Bob. This implies that although Alice has a strategy to send the right message, she does not know what this strategy is, and is thus unable to follow this strategy under imperfect information. The following example illustrates this.

Example 3.1. *Consider Protocol 3.3, in which Alice sends message m to Bob, and* NRO *and* NRR *are exchanged. First, Alice sends m and* NRO *to the TTP. The TTP forwards m to Bob, who replies by sending* NRR *and a boolean p back to the TTP. Then the TTP sends* NRO *to Bob. Alice continues by sending a boolean p' to the TTP. Only if Bob's boolean p equals Alice's boolean p', the TTP sends* NRR *to Alice.*

Protocol 3.3

1. $a \to t$: m, NRO
2. $t \to b$: m
3. $b \to t$: NRR, bool p
4. $t \to b$: NRO
5. $a \to t$: bool p'
6. If $p = p'$:
   (a) $t \to a$: NRR

Protocol 3.4

1. $t \to a$: `start`
2. $t \to b$: `start`
3. Choice:
   (a) i. $a \to t$: NRO, `request_id`
   ii. $b \to t$: NRR, id
   iii. $t \to b$: NRO
   iv. $a \to t$: `re-request_id`
   (b) i. $b \to t$: NRR, id
   ii. $a \to t$: NRO, `request_id`
   iii. $t \to b$: NRO
4. $t \to a$: NRR, id

Intuitively, Protocol 3.3 is not a fair protocol, as Alice can only obtain NRR *by sending p' in step 5 where p' equals p. However, she does not have a way of knowing p, and therefore, she does not know the correct value of p'. Nevertheless, the protocol satisfies* NEFAIR*, as $\langle\langle a_h \rangle\rangle \Diamond$NRR is true in step 5, since Alice has a correct (perfect-information) strategy: if $p =$* false*, she sends* false*, and if $p =$* true*, she sends* true*. The problem is that this strategy is not executable if Alice has imperfect information.*

In Example 3.1, it might be immediately obvious that the protocol is intuitively unfair due to problems with imperfect information. The following example is a less contrived example.

Example 3.2. *Consider Protocol 3.4 (to simplify the presentation, it is assumed that the TTP stops sending messages to agents from which he receives messages that do not correspond to the protocol). In this protocol, the non-determinism is caused by the order of arrival of messages, instead of by a boolean chosen by the other agent. In this protocol, first the TTP sends the message* `start` *to Alice and Bob. Then Alice sends* NRO *and a message* `request_id` *to request Bob's id to the TTP, and Bob sends* NRR *and his id to the TTP. However, the behavior of the TTP depends on the order in which these messages arrive. If the request arrives before the id, as in branch (a), the TTP sends* NRO *to Bob, but Alice's request is ignored until Alice sends an additional message* `re-request_id`*, on which the TTP sends her the id and* NRR*. If the request arrives after the id, as in branch (b), the TTP sends* NRO *to Bob and, immediately,* NRR *and the id to Alice.*

This implies that Alice will never obtain NRR *in the case where she does not send* `re-request_id` *in branch (a). On the other hand, in branch (b) Alice will never obtain* NRR *if she does send* `re-request_id`*. Alice cannot know or make sure that* `request_id` *arrives before or after Bob's id, and neither does she know how long the TTP will wait before answering her. Therefore, Alice does not know which branch of the protocol is executed by the TTP, which means that she does not know whether she needs to send* `request_id` *or not. Still, this protocol satisfies* NEFAIR*, as Alice has a perfect information strategy to obtain* NRR*, namely sending* `re-request_id` *in branch (a) and not sending it in (b).*

Examples 3.1 and 3.2 show that the standard interpretation of ATL* is not suitable for evaluating protocols in a context where agents might have imperfect information. This problem can be solved by interpreting ATL*-specifications in *imperfect-information concurrent game structures* [Sch04b]. In such game structures, an agent can only observe a part of the global state, and his strategy is required to choose the same action in states that he cannot distinguish. Imperfect-information concurrent game structures contain an *indistinguishability relation* $\sim_a$ for every agent $a \in \mathbb{A}\mathrm{gt}$. The indistinguishability relation connects states that agents cannot distinguish.

Definition 3.1 (Imperfect-information concurrent game structures [Sch04b])**.** *An* imperfect-information concurrent game structure *is a pair $(M, \sim)$, where M is a concurrent game structure and $\sim = (\sim_{a_1}, \sim_{a_2}, \ldots)$ such that for every agent $a \in \mathbb{A}\mathrm{gt}$, $\sim_a \in Q \times Q$ is an equivalence relation. If $q_1 \sim_a q_2$, we say that q_1 and q_2 are* indistinguishable.

We write $(q_1, \ldots, q_n) \sim_a (q'_1, \ldots, q'_n)$ whenever $q_i \sim_a q'_i$ for all i $(1 \leq i \leq n)$. Moreover, given a tuple of agents C, we write $q \sim_C q'$ whenever $q \sim_a q'$ for all agents $a \in C$.

We say that $(M, \sim_a)$ is turn-based if and only if M is turn-based.

Imperfect-information turn-based game structures are defined analogously to perfect-information turn-based game structures.

An imperfect-information strategy is required to be *uniform*, that is, if sequences of states λ, λ' are indistinguishable for agent a, written $\lambda \sim_a \lambda'$, then the strategy for agent a assigns the same action to λ and λ', i.e., $\sigma_a(\lambda) = \sigma_a(\lambda')$.

Strategies in imperfect-information concurrent game structures are required to assign the same action to states that agents cannot distinguish.

Definition 3.2 (Imperfect-information strategy)**.** *An imperfect-information strategy σ_i for agent a is a function from Q^+ to Act where $\sigma_a(q_1, \ldots, q_n) = \sigma_a(q'_1, \ldots, q'_n)$ whenever $(q_1, \ldots, q_n) \sim_a (q'_1, \ldots, q'_n)$.*

Collective imperfect-information strategies are defined analogously to perfect-information strategies.

We now interpret ATL* in imperfect-information concurrent game structures.

Definition 3.3 (Interpretation of ATL* in imperfect-information concurrent game structures [Sch04b])**.** *Let $M = ((\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta), \sim)$ be an imperfect-information concurrent game structure, let $q \in Q$ be a state, let $C \subseteq \mathbb{A}\mathrm{gt}$ be a set of*

agents, and let φ be a state formula. We write $M, q \models \langle\langle C \rangle\rangle \varphi$ if there exists a collective imperfect-information strategy σ_C such that for all agents $a \in C$, states $q' \sim_a q$, paths $\lambda \in out(q', \sigma_C)$ and positions $i \geq 0$, it holds that $M, \lambda[i, \infty] \models \varphi$. The other connectives are interpreted as in perfect-information concurrent game structures.

It should be noted that the set of uniform strategies in M is always a subset of perfect-information strategies in M. Moreover, perfect-information semantics of ATL* is well-defined in imperfect-information concurrent game structures (it simply ignores the indistinguishability relations). Furthermore, each concurrent game structure can be seen as an imperfect-information concurrent game structure where for every agent a, $\sim_a$ is the minimal reflexive relation.

We now define protocols in imperfect-information concurrent game structures. We say that an agent cannot distinguish two states if in both states, the same agent is determining the next state, and both states contain the same set of runs for that agent.

Definition 3.4 (Protocol as imperfect-information concurrent game structure)**.** *Let $M = ((\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta), \sim)$ be an imperfect-information concurrent game structure, where $(\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta)$ is the protocol interpreted as a perfect-information concurrent game structure, and $\sim = (\sim_1, \sim_2, \ldots)$ such that for agent $a \in \mathbb{A}\mathrm{gt}$, $\sim_a$ is defined as follows.*

We say that $(a_1, AKN_1, F_1) \sim_a (a_2, AKN_2, F_2)$ when either $a_1 \neq a_2$, or

$$\{((\rho, \theta, \sigma), R) \in F_1 \mid \rho(R) = a\} = \{((\rho, \theta, \sigma), R) \in F_2 \mid \rho(R) = a\}.$$

Imperfect-information semantics is sufficient to give an intuitive interpretation to StratFair, WeakFair, StrongFair and InvFair (for the latter two, the choice of semantics only matters if the initial state is unknown). However, it is not enough to "repair" NEFair. If Alice wants to be sure that she can obtain NRR, it is also necessary to use *non-enforced controled fairness* (NECFair) instead of NEFair. Non-enforced controled fairness is defined as

$$\text{NECFair} \equiv \langle\langle a_h \rangle\rangle \Box \neg (\mathsf{NRO} \wedge \neg \langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR}).$$

To see the difference between NEFair and NECFair, we define an *unfair situation* as a situation in which Bob has obtained NRO and Alice does not have a strategy to obtain NRR, i.e, $\text{Unfair} \equiv (\mathsf{NRO} \wedge \neg \langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR})$. Then we can write:

$$\begin{aligned} \text{NEFair} &\equiv \neg \langle\langle b \rangle\rangle \Diamond \text{Unfair}; \\ \text{NECFair} &\equiv \langle\langle a_h \rangle\rangle \Box \neg \text{Unfair}. \end{aligned}$$

If $\mathbb{A}\mathrm{gt} = \{a, b, t\}$ and t is deterministic, then NEFair is equivalent to the formula $\neg \langle\langle \mathbb{A}\mathrm{gt} \setminus a_h \rangle\rangle \Diamond \text{Unfair}$. This shows that NEFair and NECFair have a very similar interpretation: NEFair states that the group of agents without Alice has no common strategy to reach an unfair situation, while NECFair states that Alice has a strategy to always avoid an unfair situation.

Indeed, these two formulas are equivalent in turn-based perfect-information models [AHK02]. However, in imperfect-information models, both NEFAIR and the negation of NECFAIR can hold, as is illustrated by Protocol 3.3. On the other hand, NECFAIR does imply NEFAIR, even in imperfect-information models. In imperfect-information models, it is not sufficient that all agents but Alice have no common strategy to reach an unfair situation, as illustrated by Protocol 3.3. Therefore, in imperfect-information models, NECFAIR should be required.

From now on, we will follow Schobbens [Sch04b] and use subscripts I (respectively i) to denote that the specification is interpreted in the perfect (resp. imperfect) information semantics of ATL* whenever the type of semantics has impact on the truth of the specification. We will also write that φ_x *implies* ψ_y if and only if, for every imperfect-information concurrent game structure $M = (\mathbb{A}\mathrm{gt}, Act, Q, \Pi, \pi, d, \delta)$ and state $q \in Q$, we have that $M, q \models_x \varphi$ implies $M, q \models_y \psi$.

Fact 1. *If φ includes no strategic operators then $(\langle\!\langle C \rangle\!\rangle \varphi)_i$ implies $(\langle\!\langle C \rangle\!\rangle \varphi)_I$. The converse does not hold in general.*

3.5 Effective Fairness

Fairness is not a sufficient property for fair-exchange protocols. Fair-exchange protocols should not only satisfy fairness, but also *effectiveness* (sometimes also called *viability*). It turns out that combining fairness and effectiveness is not trivial. The second limitation with the current way of modeling fairness in ATL* has to do with this combination.

To see the need for effectiveness, consider the *empty protocol*, i.e., the (admittedly useless) protocol, that specifies that no messages are sent. It is obvious that this protocol satisfies all definitions of fairness discussed above, as no unfair situation can possibly occur. Still, the protocol is clearly not a good fair-exchange protocol, because even if the agents want to, they cannot exchange evidences.

To exclude protocols like this, we need to impose a second property (besides fairness), that states that the protocol is *effective*. This property states that Alice and Bob have a collective strategy to execute the protocol such that both agents obtain their evidence. Effectiveness can be formulated in ATL* as follows:

$$\textsc{Effective} \equiv \langle\!\langle a_h, b_h \rangle\!\rangle \Diamond (\mathsf{NRO} \wedge \mathsf{NRR}).$$

Requiring effectiveness excludes the empty protocol. However, requiring both effectiveness and non-enforced fairness is not sufficient to rule out bad protocols. To see this, consider the following example.

Example 3.3. *Let us consider Protocol 3.5.*

Protocol 3.5

1. Choice for a:
   (a) i. $a \to b$: NRO
   ii. $b \to a$: NRR
   (b) End of protocol.

In this protocol, Alice can choose to either send NRO *to Bob and wait for* NRR *to be sent to her, or immediately stop the protocol.*

This protocol is effective (if Alice chooses 1a and both parties continue the protocol, they get their evidence). Furthermore, the protocol satisfies non-enforced fairness, because Alice has a strategy to achieve fairness (by choosing 1b). Thus, the protocol satisfies both (non-enforced) fairness and effectiveness. However, intuitively, it is still not a good protocol, as Bob might be dishonest and stop the protocol after 1(a)i, leaving Alice without her evidence.

The problem discussed in Example 3.3 arises because a's strategy that guarantees effectiveness is different from a's strategy that guarantees fairness. To solve this problem, we need to require that there exists a strategy for Alice that satisfies both effectiveness and fairness at the same time. This cannot be easily expressed in ATL*. One way to overcome this problem is that, at least in turn-based game structures, it is sufficient to require that Alice and Bob guarantee fairness until NRO and NRR have been exchanged. In perfect-information turn-based game structures, this property can be expressed in ATL* as follows:

$$\langle\langle a_h, b_h \rangle\rangle(\text{NECFAIR } \mathcal{U} (\mathsf{NRO} \wedge \mathsf{NRR})).$$

This formula expresses that a and b have a collective strategy that guarantees NECFAIR for Alice until both Bob and Alice have their evidence.

The formula requires that Bob is honest in the outer quantifier, but allows Bob to be dishonest in the quantifier inside NEFAIR. This is a problem, as agents need to be either modeled as honest or dishonest. Therefore, we introduce an additional proposition Honest_b, which is true as long as Bob has only sent messages allowed by the protocol. Now we can reformulate the property for Bob's honesty so that it applies only to effectiveness and not fairness:

$$\text{EFFFAIR} \equiv \langle\langle a_h, b \rangle\rangle(\text{NECFAIR } \mathcal{U} (\mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b)).$$

Now we show that effective fairness indeed guarantees both effectiveness and non-enforced fairness.

Theorem 3.1. EFFFAIR_I *implies* EFFECTIVE_I *and* NECFAIR_I*, and* EFFFAIR_i *implies* EFFECTIVE_i *and* NECFAIR_i.

Proof. That EFFFAIR_I implies EFFECTIVE_I and that EFFFAIR_i implies EFFECTIVE_i follows directly.

To show that EffFair$_I$ implies NECFair$_I$, we assume $M, q \models$ EffFair$_I$, and show that NECFair$_I$. First we assume that $M, q \models_I \mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$. Let λ be a q-path, and let $i \geq 0$. Then we have $M, \lambda[0] \models_I \mathsf{NRR}$, and therefore also $M, \lambda[i] \models_I \mathsf{NRR}$ (as NRR is a property that stays true after it has been true for the first time). Therefore, it holds that $M, \lambda[i] \models_I \langle\langle a_h \rangle\rangle(\top \mathcal{U} \mathsf{NRR})$ and thus that $M, \lambda[i] \models_I \langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR}$, and therefore $M, \lambda[i] \not\models_I \mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR}$, which implies $M, \lambda[i] \models_I \neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR})$. This implies $M, \lambda \models_I \Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR})$, and thus, it holds that $M, q \models_I \langle\langle a_h \rangle\rangle \Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR}) =$ NECFair. Now we assume that $M, q \not\models \mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$. Then $M, q \models_I$ EffFair$_I = \langle\langle a_h, b \rangle\rangle$(NECFair $\mathcal{U}$ ($\mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$). Since $M, q \not\models \mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$, we have $M, q \models_I$ NECFair.

To show that EffFair$_i$ implies NECFair$_i$, we assume $M, q \models$ EffFair$_i$, and show that NECFair$_i$. First we assume that for all $q' \sim_a q$, we have $M, q' \models_i \mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$. Let $q' \sim_a q$, let λ be a q'-path, and let $i \geq 0$. Then we have $M, \lambda[0] \models_i \mathsf{NRR}$, and therefore also $M, \lambda[i] \models_i \mathsf{NRR}$ (as NRR is a property that stays true after it has been true for the first time). Then it holds that $M, \lambda[i] \models_i \langle\langle a_h \rangle\rangle(\top \mathcal{U} \mathsf{NRR})$ and thus $M, \lambda[i] \models_i \langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR}$, and therefore $M, \lambda[i] \not\models_i \mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR}$, which implies $M, \lambda[i] \models_i \neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR})$. Therefore, we have $M, \lambda \models_i \Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR})$, and thus, it holds that $M, q \models_i \langle\langle a_h \rangle\rangle \Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR}) =$ NECFair. Now we assume that not for all $q' \sim_a q$, we have $M, q \models \mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$. Let $q' \sim_a q$ such that $M, q' \not\models \mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$, and let λ be a q'-path. Since $q' \sim_a q$, we have $q' \sim_{\{a,b\}} q$, so by $M, q \models_i$ EffFair$_i$, we have $M, \lambda \models_i \langle\langle a_h, b \rangle\rangle$(NECFair $\mathcal{U}$ ($\mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$)). Since $M, q' \not\models_i \mathsf{NRO} \wedge \mathsf{NRR} \wedge \text{Honest}_b$, we have $M, q' \models_i$ NECFair. Then for every q''-path λ where $q'' \sim_a q'$, we have $M, \lambda \models \Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR})$. Therefore, since $\sim_a$ is an equivalence relation, for every q''-path λ where $q'' \sim_a q$, we have $M, \lambda \models \Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h \rangle\rangle \Diamond \mathsf{NRR})$ as well. This implies that $M, q \models$ NECFair$_i$. □

Note that the converse implications do not hold. For example, Protocol 3.5 satisfies both Effective and NEFair, but not EffFair.

We observe that EffFair$_I$ suffers from the problems concerning imperfect information mentioned in Section 3.4. Moreover, even if a protocol satisfies EffFair$_i$, it can still be the case that the strategies for Alice behind the outer and the nested strategic operators cannot be combined into a single uniform strategy (cf. [JB11]). Consider the situation where Alice can either stop, resulting in fairness but not effectiveness, or continue, only resulting in fairness (and effectiveness) if Bob plays honest and neither fairness nor effectiveness otherwise. This is problematic if Alice does not know whether Bob plays honest: in that case, EffFair$_I$ is satisfied, but Alice does not have a strategy that results in both fairness and effectiveness.

We have shown that fairness is not sufficient for fair-exchange protocols, and that effectiveness is also needed. Moreover, non-enforced fairness and effectiveness cannot be combined trivially. We give a new specification, EffFair, that handles this combination. This problem does not occur for weak, strategic, strong or invariant fairness and effectiveness. For these specifications, it is sufficient to require the conjunction of fairness and effectiveness.

Example 3.4. Fair-exchange protocols *are security protocols in which two agents, which we call Alice and Bob, want to fairly exchange digital items, for example*

signatures on a contract [Zho10]. We write p_A when Alice gets Bob's item, and p_B when Bob gets Alice's item. We say that a protocol satisfies fairness *for Alice, if Alice can guarantee that if Bob gets the item, Alice will get it as well. Fairness for Bob, denoted φ_B, is defined symmetrically. Furthermore, we say that a protocol is effectiveness, denoted φ_C, if Alice and Bob can guarantee that they both get the item.*

In order for a fair-exchange protocol to be correct, the protocol must satisfy effective fairness*, i.e., the strategy with which an agent guarantees fairness must be the same strategy as with which he guarantees effectiveness [CR10, JMM12]. Formally, the strategies for both objectives must be the same, i.e., there exists a pair of strategies (σ_A, σ_B) such that:*

1. *If Alice plays σ_A, then her goal φ_A is achieved;*
2. *If Bob plays σ_B, then his goal φ_B is achieved;*
3. *If Alice plays σ_A and Bob plays σ_B, then the common goal φ_C is achieved.*

3.6 Hierarchy of Fairness Requirements

We proceed by studying the relations between the different definitions of fairness. Figure 3.1 contains a graphical view of these relations. Below we include proofs for some of the relations. The other cases are straightforward. Unless explicitly stated otherwise, the same reasoning applies to both the perfect-information and imperfect-information version of variants of ATL*.

Strong, Strategic, Weak and Invariant Fairness Chadha et al. [CKS06] prove that

$$\textsc{StrongFair}_I \Rightarrow \textsc{StratFair}_I \Rightarrow \textsc{WeakFair}_I \Rightarrow \textsc{InvFair}_I.$$

The two final implications extend to imperfect information. Furthermore, Chadha et al. [CKS06] show that $\textsc{StratFair}_I$, $\textsc{WeakFair}_I$ and $\textsc{InvFair}_I$ are equivalent under *strategic timeliness*. Strategic timeliness states that Alice always has an honest strategy that eventually allows her to stop executing the protocol: $\textsc{Timely} \equiv \forall\Box(\langle\!\langle a_h \rangle\!\rangle \Diamond \mathrm{Stop}_a)$. Furthermore, $\textsc{InvFair}_I$, $\textsc{WeakFair}_I$ and $\textsc{StrongFair}_I$ are clearly equivalent with $\textsc{InvFair}_i$, $\textsc{WeakFair}_i$ and $\textsc{StrongFair}_i$, respectively, as they do not contain strategic modalities.

These are the only implications that hold between StrongFair, StratFair, WeakFair and InvFair. We show this by providing a number of counterexamples, as displayed in Figure 3.2. Protocol 3.6 satisfies StratFair, but not StrongFair. Protocol 3.7 (a protocol lacking strategic timeliness) satisfies WeakFair but not StratFair. Protocol 3.8 (another protocol lacking strategic timeliness) satisfies InvFair, but not WeakFair. Finally, neither $\textsc{StrongFair}_i \Rightarrow \textsc{StratFair}_i$ nor $\textsc{WeakFair}_i \Rightarrow \textsc{StratFair}_i$ is valid, even under strategic timeliness, as in a model where the initial state with $\neg\mathsf{NRO}$ is indistinguishable from an unreachable state with $\mathsf{NRO}\wedge\neg\mathsf{NRR}$, we might have $\textsc{StrongFair}_i$ and $\textsc{WeakFair}_i$, but not $\textsc{StratFair}_i$.

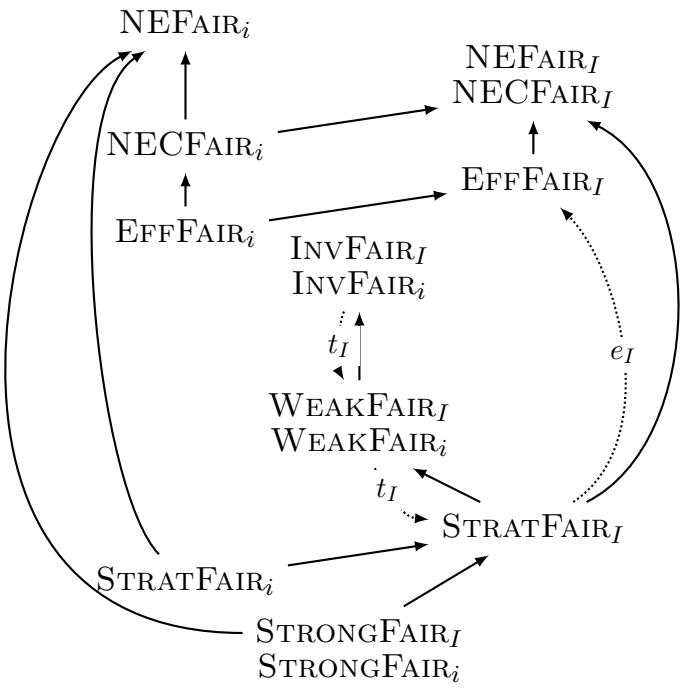


Figure 3.1: Relations between different notions of fairness. Solid arrows stand for implications, i.e., lead from stronger to weaker definitions of fairness. Dashed arrows represent implications that hold only under additional assumptions of effectiveness (e) or strategic timeliness (t). Missing arrows correspond to implications that do not hold. *Note:* we did not include arrows that follow from transitivity of implication.

Non-enforced Fairness Now we study how StrongFair, StratFair, WeakFair and InvFair relate to NEFair.

Theorem 3.2. StratFair *implies* NEFair.

Proof. Assume $\forall\Box(\mathsf{NRO} \to \langle\langle a_h\rangle\rangle\Diamond\mathsf{NRR})$. Because $\forall\Box\varphi \equiv \neg\exists\Diamond\neg\varphi$ is a validity of CTL, we have $\neg\exists\Diamond(\mathsf{NRO} \wedge \neg\langle\langle a_h\rangle\rangle\Diamond\mathsf{NRR})$. Therefore, it holds that $\neg\langle\langle b\rangle\rangle\Diamond(\mathsf{NRO} \wedge \neg\langle\langle a_h\rangle\rangle\Diamond\mathsf{NRR})$. □

Similarly, StrongFair implies NEFair as well. Also, because specifications StratFair, WeakFair and InvFair are equivalent given strategic timeliness, WeakFair and InvFair imply NEFair given strategic timeliness. Now we show that the other implications do not hold. Protocol 3.9 satisfies NEFair, but not StrongFair, StratFair, WeakFair or InvFair. Protocol 3.7, a protocol that does not satisfy strategic timeliness, satisfies InvFair and WeakFair, but not NEFair. Finally, StratFair$_i \Rightarrow$ NECFair$_i$ is not valid, as it does not hold in a model with a state q with a next state where $\mathsf{NRO} \wedge \neg\mathsf{NRR}$ holds such that q is indistinguishable from the initial state.

Moreover, as shown in Section 3.4, NEFair and NECFair are equivalent under perfect information, while under imperfect information, NECFair implies NEFair, but not vice versa.

Protocol 3.6

1. $b \to t$: NRR
2. $t \to a$: `continue`
3. $a \to b$: NRO
4. Choice for a:
   (a) i. $a \to t$: true
   ii. $t \to a$: NRR
   (b) i. $a \to t$: false
   ii. Go to 4.

Protocol 3.7

1. $a \to b$: NRO
2. Choice for b:
   (a) i. $b \to a$: NRR
   (b) i. $b \to a$: `cont.`
   ii. Go to 2.

Protocol 3.8

1. $a \to b$: NRO
2. $b \to a$: `continue`
3. Go to 2.

Protocol 3.9

1. $b \to t$: NRR
2. $a \to b$: NRO
3. Choice for a:
   (a) i. End of protocol.
   (b) i. $a \to t$: `cont.`
   ii. $t \to a$: NRO

Figure 3.2: Counterexample protocols

Effective Fairness We proceed by studying the relations between EffFair and the other definitions of fairness. EffFair implies NEFair, as shown in Theorem 3.1. The following theorem states that in effective protocols, StratFair$_I$ implies EffFair$_I$. This theorem does only hold assuming perfect information. Under imperfect information, Alice is not guaranteed to know whether Bob plays honest, and cannot decide whether she should continue the cooperation with Bob or not.

Theorem 3.3. *Whenever* Effective$_I$ *holds,* StratFair$_I$ *implies* EffFair$_I$.

Proof. Assume that Effective$_I$ and StratFair$_I$ hold. We set $\varphi = \neg(\mathsf{NRO} \wedge \neg\langle\langle a_h\rangle\rangle\Diamond\mathsf{NRR})$ and $\psi = \mathsf{NRO} \wedge \mathsf{NRR}$. StratFair$_I$ $= \forall\Box(\mathsf{NRO} \to \langle\langle a_h\rangle\rangle\Diamond\mathsf{NRR})_I$ is equivalent to $\forall\Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h\rangle\rangle\Diamond\mathsf{NRR})_I$ and can thus be written as $(\forall\Box\varphi)_I$. This means that for all paths $\lambda \in out(q, \emptyset)$ and all positions $i \geq 0$, we have $\lambda[i] \models_I \forall\Box\varphi$ as well (1). Effective$_I$ can be written as $(\langle\langle a_h, b_h\rangle\rangle\Diamond\psi)_I$. By definition of $\Diamond$, there exists a pair F of strategies for agents a_h and b_h, respectively, such that for all $\lambda \in out(q, F)$ there exists $i \geq 0$ with $\lambda[i] \models_I \psi$ (2). Let F be a pair of strategies for a and b satisfying this condition. Then we have that for all $\lambda \in out(q, F)$ there exists $i \geq 0$ with $\lambda[i] \models_I \psi$ by (2), and for all $0 \leq j < i$, we have $\lambda[j] \models_I \langle\langle a_h\rangle\rangle\Box\varphi$ by (1). By definition of $\mathcal{U}$, we obtain $q \models_I \langle\langle a_h, b_h\rangle\rangle((\langle\langle a_h\rangle\rangle\Box\neg(\mathsf{NRO} \wedge \neg\langle\langle a_h\rangle\rangle\Diamond\mathsf{NRR}))\ \mathcal{U}\ (\mathsf{NRO} \wedge \mathsf{NRR}))$, i.e., EffFair$_I$. □

Again, these results, and the transitive closures of them, are all the implications that hold. Protocol 3.5 satisfies NEFair, but not EffFair. Furthermore, the empty protocol, which obviously does not satisfy effectiveness, satisfies StrongFair, StratFair, WeakFair and InvFair, but not EffFair. Finally, Protocol

3.7, not satisfying strategic timeliness, satisfies WeakFair and InvFair, but not EffFair.

3.7 Related Work

Various definitions of non-repudiation and fair exchange have been formalized. However, as we argue in this chapter, these definitions are often either too strong or too weak because they do not take into account the agents' ability to choose the right strategy. In this section, we discuss how our results relate to existing proposals about verification of non-repudiation protocols and other fair-exchange protocols with the strategic logic ATL*.

Kremer and Raskin [KR03] use NEFair to verify various non-repudiation protocols. They find flaws in the Zhou-Gollmann optimistic protocol [ZG97a], the Asokan-Shoup-Waidner certified mail protocol [ASW98], and the Markowitch–Kremer multi-party non-repudiation protocol [MK00]. An improved version of the latter protocol, as well as the Kremer-Markowitch non-repudiation protocol [KM00], is shown to satisfy NEFair. However, as we have seen in Section 3.4, the protocols that are shown to satisfy NEFair might still be unfair if the agents' strategies are not executable due to imperfect information. Furthermore, all strategies that guarantee fairness in these protocols might be ineffective, as we showed in Section 3.5.

Chadha et al. [CKS06] demonstrate that the GM protocol [GM99], a multi-party contract signing protocol, does not satisfy InvFair, WeakFair, StratFair and StrongFair for four participants. However, as we have seen, non-enforced fairness might still hold. It can be argued that non-enforced fairness is sufficient, if it is assumed that Alice has the ability to resolve the choices in a non-deterministic protocol in the way that is the most advantageous for her.

Liu et al. [LPZ11] propose an extended CEM (certified e-mail) protocol with TTP transparency and use StratFair to prove fairness. However, strategic timeliness is only checked in a perfect information model, which means that the protocol may be intuitively unfair in the presence of imperfect information, as we saw in Section 3.4. Furthermore, the extended CEM protocol does not necessarily have strong fairness, as StratFair does not imply StrongFair. This means that it is still important that the agents resolve the non-determinism of the protocol in the correct way.

Finally, Zhang et al. [ZZPM09] analyze a number of multi-party contract signing protocols. WeakFair and InvFair are used to prove that the MR protocol [MR08] is fair with up to 5 signers, and that the MRT protocol [MRD09] with 3 signers has a flaw. Furthermore, a corrected MRT protocol for 3 and 4 signers is presented, which is shown to satisfy WeakFair and InvFair. Because strategic timeliness is proven, the results carry over to StratFair. We saw in Section 3.6 that StratFair does not imply StrongFair, and that NEFair does not imply StratFair. Therefore, it could be that both the original and the corrected version of the MRT protocol satisfy NEFair, i.e., are fair assuming agents have enough rationality to take the correct choices. On the other hand, it could be that both the original and corrected version of the MRT protocol lack StrongFair, i.e., that in

both protocols, not every way of resolving non-determinism leads to fairness. In the same way, the successful verification of StratFair in the MR protocol does not guarantee NEFair. Furthermore, as strategic timeliness is only checked in a perfect information model, the MR protocol and the corrected MRT protocol might be only fair under the unrealistic assumption of perfect information (see Section 3.6).

3.8 Conclusion

We have shown that the specifications of fairness that are currently used for the verification of non-repudiation and other fair-exchange protocols have a number of limitations. First, one of the definitions of fairness, non-enforced fairness, accepts intuitively unfair protocols, because it was not taken into account that agents can have imperfect information. This makes it clear that formal verification should take imperfect information into account. We have proposed a new definition of fairness that can be used in models with imperfect information. Furthermore, we have shown that fairness is not a sufficient property for fair-exchange protocols, as protocols are also required to be effective. We have shown that if both fairness and effectiveness are expressed in terms of strategies, the two properties cannot be combined easily in ATL*. We have proposed a new definition of fairness that combines the properties correctly in turn-based games. Moreover, we have given a hierarchy of the various definitions of fairness, and have proven that this hierarchy is correct. Finally, we have shown that our results have consequences for existing results from literature.

Which definition of fairness is most appropriate depends on the assumptions on the agents. If the agents are not rational and should be protected against taking bad decisions, then StrongFair is clearly the best option. If the agents are rational, the situation is more sophisticated, as we know how to specify fairness and effectiveness under imperfect information but *not* both at the same time. To find as many flaws as possible, we recommend to verify EffFair in imperfect information semantics. However, even protocols that satisfy this specification might be flawed: in imperfect-information models, EffFair guarantees the existence of a strategy that is fair, and the existence of a strategy that is effective, but not the existence of a strategy that is both fair and effective. In Chapter 4, we will show that such a specification cannot be expressed in ATL* in imperfect-information models.

For future work, it would be interesting to study ATL* specifications of *abuse-freeness*, a property that guarantees that no signer can prove to an outside observer that he is able to determine the result of the protocol. Moreover, it would be useful to have a tool to verify the concepts of fairness for existing non-repudiation protocols. This may require a fundamental extension of verification techniques as there are no ATL* model checkers for imperfect information. There was an attempt in one of the older versions of MCMAS [LQR09], but because of conceptual as well as computational problems the extension was subsequently abandoned. Also, the Alpaga tool [BCW$^+$09] can only solve a limited fragment of imperfect information games.

4

The Expressive Power of ATL*

Abstract. We compare the expressive power of ATL* and the simple fragment of Strategy Logic. We show that on turn-based game structures, ATL* is at least as expressive as the unnested simple one-alternating fragment of Strategy Logic without $\bigcirc$ operator. We do so by providing a translation from this fragment into ATL*. On concurrent game structures, however, we show that the simple fragment of Strategy Logic is in fact more expressive than ATL*. This is also the case on imperfect-information game structures, even when they are turn-based. These results imply that ATL* is not expressive enough to express the combination of fairness and effectiveness in neither turn-based models nor imperfect-information models.

4.1 Introduction

In Chapter 3, we saw that it was difficult to combine the requirements of fairness and effectiveness in turn-based and imperfect-information models in ATL*. In this chapter, we formally show that doing so is impossible. We do so by expressing the combination of fairness and effectiveness in Strategy Logic, and formally comparing the expressive power of ATL* and SSL. The results of this comparison are as follows. First, we point out a problem in the proof of [CHP10] that SSL is as expressive as ATL* in turn-based game structures. However, we show that his result is still valid for a fragment of SSL. Furthermore, we show that SSL is more expressive than ATL* in *concurrent* game structures, and give a characterization of SSL-formulas that can be expressed in ATL*. We do this by extending the definition of ATL*-bisimulation from [ÅGJ07, DJ10] to imperfect information. Finally, we show that in turn-based game structures, SSL is more expressive than ATL*. We conclude by applying these results back to security properties. In particular, we show that the property that expresses the combination of fairness and effectiveness falls outside the class of SSL formulas that can be expressed in ATL*. This implies that the combination of fairness and effectiveness cannot be expressed in ATL*. All results in this chapter hold for both finite and infinite concurrent game structures.

4.2 Preliminaries

4.2.1 Expressive Power

We start by defining what we mean by *expressive power* [WD09]. We say that language $\mathcal{L}_2$ is *at least as expressive* as language $\mathcal{L}_1$ if for every formula in $\mathcal{L}_1$,

there exists a formula in $\mathcal{L}_2$ such that both formulas agree on all models.

Definition 4.1 (Expressive power)**.** *Language $\mathcal{L}_2$ is at least as expressive as language $\mathcal{L}_1$ on a class of models $\mathcal{M}$ with satisfiability relation $\models$, written $\mathcal{L}_1 \sqsubseteq_{\mathcal{M}} \mathcal{L}_2$, if*

$$\forall \varphi_1 \in \mathcal{L}_1.\ \exists \varphi_2 \in \mathcal{L}_2.\ \forall M \in \mathcal{M}.\ M \models \varphi_1 \text{ if and only if } M \models \varphi_2.$$

We say that $\mathcal{L}_1$ and $\mathcal{L}_2$ are equally expressive *on class of models $\mathcal{M}$, written $\mathcal{L}_1 \equiv_{\mathcal{M}} \mathcal{L}_2$, if $\mathcal{L}_1 \sqsubseteq_{\mathcal{M}} \mathcal{L}_2$ and $\mathcal{L}_2 \sqsubseteq_{\mathcal{M}} \mathcal{L}_1$. Finally, we say that $\mathcal{L}_2$ is* more expressive *than $\mathcal{L}_1$ on class of models $\mathcal{M}$, written $\mathcal{L}_1 \sqsubset_{\mathcal{M}} \mathcal{L}_2$, if $\mathcal{L}_1 \sqsubseteq_{\mathcal{M}} \mathcal{L}_2$ and not $\mathcal{L}_1 \equiv_{\mathcal{M}} \mathcal{L}_2$.*

Example 4.1. *Let $\mathrm{PROP}_{\mathcal{C}}$ be propositional logic with only connectives $\mathcal{C}$, and let $\mathcal{M}$ be a valuation that assigns truth values to atomic formulas (the usual model for propositional logic). Then we have $\mathrm{PROP}_{\{\vee,\wedge\}} \sqsubseteq_{\mathcal{M}} \mathrm{PROP}_{\{\vee,\wedge,\neg\}}$, $\mathrm{PROP}_{\{\vee,\wedge,\neg\}} \equiv_{\mathcal{M}} \mathrm{PROP}_{\{\vee,\neg\}}$, and $\mathrm{PROP}_{\{\vee,\wedge\}} \sqsubset_{\mathcal{M}} \mathrm{PROP}_{\{\vee,\wedge,\neg\}}$.*

4.2.2 The Simple One-alternating Fragment of Strategy Logic

In many domains, it is desirable to write specifications that treat strategies in two-agent games as explicit first-order objects. Such domains include assume-guarantee synthesis [CR10], secure equilibria [CHJ06], and verification of fair-exchange protocols [CR10, JMM12].

Several logics have been proposed that quantify explicitly over strategies. These include Strategy Logic (SL) as defined by Chatterjee, Henzinger & Piterman [CHP10], and a logic defined by Mogavero et al. [MMPV11] also named Strategy Logic ($\mathrm{SL}_{\mathrm{MMV}}$).

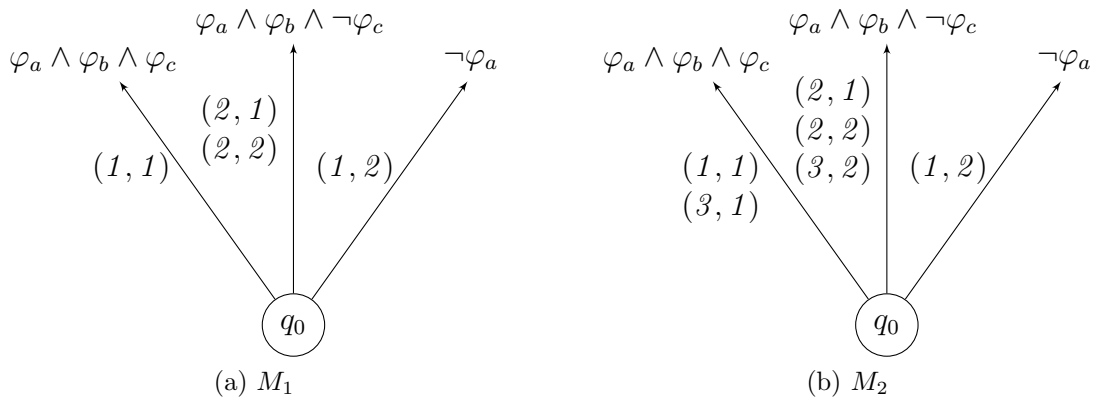
In this thesis, we focus on the simple one-alternating fragment of Strategy Logic (SSL) [CHP10], which is strictly contained in both SL and $\mathrm{SL}_{\mathrm{MMV}}$. SSL is strong enough to model many interesting properties. In particular, we will use SSL to express the combination of fairness and effectiveness.

The version of Strategy Logic as defined in [CHP10] is interpreted over turn-based game structures. We extend this definition to concurrent game structures. We only give the definition of SSL for games with two agents, a and b, but the definition can be easily extended to multiple agents. The crucial operator of SSL is $(\exists\ \varphi_a, \exists\ \varphi_b, \varphi_c)$. This operator expresses that a and b have a joint strategy to achieve the common goal φ_c; moreover, if a plays his part of the joint strategy, then φ_a is achieved independently of the behavior of b, and vice versa.

Definition 4.2 (Simple one-alternating fragment of Strategy Logic (SSL))**.** *The simple one-alternating fragment of Strategy Logic (*SSL*) is defined as the following set Φ, where Π is a set of propositions:*

$$\Phi ::= \Pi \mid \neg\Phi \mid \Phi \wedge \Phi \mid (\exists\ \Psi, \exists\ \Psi, \Psi);$$
$$\Psi ::= \Phi \mid \neg\Psi \mid \Psi \vee \Psi \mid \bigcirc\Psi \mid \Box\Psi \mid \Psi\ \mathcal{U}\ \Psi.$$

We proceed by giving the interpretation of SSL in concurrent game structures.



(a) M_1 (b) M_2

Figure 4.1

Definition 4.3 (Interpretation of SSL)**.** *Given an* SSL *formula φ, a concurrent game structure M, and a state q, the interpretation $M, q \models \varphi$ is defined as follows.*

$$M, q \models \neg\varphi \text{ if } M, q \not\models \varphi;$$
$$M, q \models \varphi_1 \wedge \varphi_2 \text{ if } M, q \models \varphi_1 \text{ and } M, q \models \varphi_2;$$
$$M, q \models (\exists\, \varphi_a, \exists\, \varphi_b, \varphi_c) \text{ if there exist strategies } \sigma_a, \sigma_b \text{ such that:}$$

- *For all paths $\lambda \in out(q', \sigma_a)$, we have $M, \lambda \models \varphi_a$; and*
- *For all paths $\lambda \in out(q, \sigma_b)$, we have $M, \lambda \models \varphi_b$; and*
- *For all paths $\lambda \in out(q, (\sigma_a, \sigma_b))$, we have $M, \lambda \models \varphi_c$.*

The path formulas Ψ are defined as in ATL*.

Example 4.2. *Consider concurrent game structures M_1 and M_2 from Figure 4.1. We have $M_1, q_0 \not\models (\exists \Diamond\varphi_a, \exists \Diamond\varphi_b, \Diamond\varphi_c)$, but $M_2, q_0 \models (\exists \Diamond\varphi_a, \exists \Diamond\varphi_b, \Diamond\varphi_c)$.*

4.2.3 Strategy Logic for Imperfect Information

The original version of Strategy Logic [CHP10] is only defined for perfect-information concurrent game structures. We propose an extension of the simple one-alternating fragment of Strategy Logic to imperfect-information concurrent game structures [Sch04b].

Now we propose an interpretation of SSL in imperfect-information concurrent game structures.

Definition 4.4 (Interpretation of SSL)**.** SSL *formulas are interpreted in an imperfect-information concurrent game structure M and a state q in the following way.*

$$M, q \models \varphi_1 \wedge \varphi_2 \text{ if } M, q \models \varphi_1 \text{ and } M, q \models \varphi_2;$$
$$M, q \models \varphi_1 \vee \varphi_2 \text{ if } M, q \models \varphi_1 \text{ or } M, q \models \varphi_2;$$
$$M, q \models (\exists\, \varphi_a, \exists\, \varphi_b, \varphi_c) \text{ if there exist imperfect-information strategies } \sigma_a, \sigma_b$$

such that:

- *for all states q' such that $q' \sim_a q$ and paths $\lambda \in out(q', \sigma_a)$, we have $M, \lambda \models \varphi_a$; and*
- *for all states q' such that $q' \sim_b q$ and paths $\lambda \in out(q, \sigma_b)$, we have $M, \lambda \models \varphi_b$; and*
- *for all states q' such that $q' \sim_a q$ or $q' \sim_b q$, and all paths $\lambda \in out(q, (\sigma_a, \sigma_b))$, we have $M, \lambda \models \varphi_c$.*

Note that concurrent game structures can be seen as special cases of imperfect-information concurrent game structures, namely those where $\sim_a$ is the identity relation for all agents a.

4.3 Expressive Power in Turn-based Models

It is claimed in Chatterjee et al. [CHP10] that ATL* is at least as expressive as SSL. However, the proof in that paper uses incomplete case distinction. It is stated that every path formula on a labeled game graph can be reduced to an infinitary condition, and that Strategy Logic with infinitary conditions can be modeled in ATL*. However, path formulas can only be reduced to infinitary conditions on a different game graph.

We show that ATL* is at least as expressive as the *unnested $\bigcirc$-free* fragment of SSL. However, we leave open the general question whether ATL* is at least as expressive as SSL.

4.3.1 The Unnested $\bigcirc$-free Fragment of SSL

We define $\text{SSL}^{+\oslash}$, the unnested $\bigcirc$-free fragment of SSL, as the fragment of SSL without $\bigcirc$ where every path formula contains exactly one temporal operator.

Definition 4.5 ($\text{SSL}^{+\oslash}$)**.** $\text{SSL}^{+\oslash}$*, the unnested $\bigcirc$-free fragment of* SSL *is defined as the following set Φ.*

$$\Phi ::= \Pi \mid \Phi \wedge \Phi \mid \Phi \vee \Phi \mid (\exists\ \Psi, \exists\ \Psi, \Psi),$$
$$\Psi ::= \Phi \mid \neg\Psi \mid \Psi \vee \Psi \mid \Box\Phi \mid \Phi\ \mathcal{U}\ \Phi.$$

4.3.2 Weak-until Positive Normal Form

We show that ATL* is as expressive as the unnested SSL-free fragment, by giving a translation of SSL formulas to equivalent ATL* formulas. The first step of this translation consists of writing the formulas in *weak-until positive normal form* [BK08]. In this normal form, we add an additional operator $\mathcal{W}$. The formulas $\varphi\ \mathcal{U}\ \psi$ and $\varphi\ \mathcal{W}\ \psi$ both express that φ is true until ψ is true. However, $\varphi\ \mathcal{U}\ \psi$ requires that ψ eventually will be true, while in $\varphi\ \mathcal{W}\ \psi$, it might be the case that ψ remains false forever.

Definition 4.6 (Weak until [BK08])**.** *We define $\mathcal{W}$ (*weak until*) by $\varphi\ \mathcal{W}\ \psi ::= (\varphi\ \mathcal{U}\ \psi) \vee \Box\varphi$.*

In weak-until positive normal form, negation occurs only directly in front of state formulas, and $\Box$ is replaced by $\mathcal{W}$. Every path formula can be translated into positive normal form [BK08].

Definition 4.7 (Weak-until positive normal form [BK08])**.** *The set Ψ of unnested $\bigcirc$-free path formulas in weak normal form is defined as follows:*

$$\Phi ::= \Pi \mid \neg\Phi \mid \Phi \wedge \Phi \mid (\exists\ \Psi, \exists\ \Psi, \Psi);$$
$$\Psi ::= \Phi \mid \neg\Phi \mid \Psi \vee \Psi \mid \Psi \wedge \Psi \mid \Phi\ \mathcal{U}\ \Phi \mid \Phi\ \mathcal{W}\ \Phi.$$

4.3.3 Translation

In order to define the translation from SSL formulas to equivalent ATL* formulas, we define a set of *relevant* formulas $\mathsf{Relv}(\varphi)$ given a formula φ. The idea behind this set is that to determine the validity of $\varphi_1\ \mathcal{W}\ \varphi_2$ or $\varphi_1\ \mathcal{U}\ \varphi_2$ in a path, one only needs to determine whether φ_1 is false somewhere in the path, and whether φ_2 is true somewhere in the path. Therefore, in $\varphi_1\ \mathcal{W}\ \varphi_2$ and $\varphi_1\ \mathcal{U}\ \varphi_2$, only the occurrence of $\neg\varphi_1$ and φ_2 in the path are relevant.

Definition 4.8 (Relevant)**.** *The relevant formulas $\mathsf{Relv}(\psi)$ of a path formula ψ are recursively defined on the structure of ψ in the following way,*

$$\begin{aligned}
\mathsf{Relv}(\psi) &= \emptyset \\
\mathsf{Relv}(\neg\psi) &= \emptyset \\
\mathsf{Relv}(\psi_1 \vee \psi_2) &= \mathsf{Relv}(\psi_1) \cup \mathsf{Relv}(\psi_2) \\
\mathsf{Relv}(\psi_1 \wedge \psi_2) &= \mathsf{Relv}(\psi_1) \cup \mathsf{Relv}(\psi_2) \\
\mathsf{Relv}(\psi_1\ \mathcal{U}\ \psi_2) &= \{\neg\psi_1, \psi_2\} \\
\mathsf{Relv}(\psi_1\ \mathcal{W}\ \psi_2) &= \{\neg\psi_1, \psi_2\}
\end{aligned}$$

Example 4.3. *The formulas $\Diamond p$ and $\Box q$ are equivalent to $\top\ \mathcal{U}\ p$ and $q\ \mathcal{W}\ \bot$, respectively. So the relevant formulas in $\Diamond p \wedge \Box q$ are $\mathsf{Relv}((\top\ \mathcal{U}\ p) \wedge (q\ \mathcal{W}\ \bot)) = \{\neg\top, p, \neg q, \bot\}$.*

We proceed by defining the *goal* φ^G of formula φ after occurrence of a set of formulas G. The intuition behind the definition of $(\psi_1\ \mathcal{U}\ \psi_2)^G$ and $(\psi_1\ \mathcal{W}\ \psi_2)^G$ is as follows. In a path with a state where ψ_2 is satisfied in an earlier state, $\psi_1\ \mathcal{U}\ \psi_2$ and $\psi_1\ \mathcal{W}\ \psi_2$ are satisfied. In a path with a state where $\neg\psi_1$ is satisfied in an earlier state while ψ_2 is not satisfied in any earlier state, $\psi_1\ \mathcal{U}\ \psi_2$ and $\psi_1\ \mathcal{W}\ \psi_2$ are not satisfied. Otherwise, i.e., in a path with a state where neither ψ_1 nor ψ_2 has been satisfied in an earlier state, $\psi_1\ \mathcal{U}\ \psi_2$ and $\psi_1\ \mathcal{W}\ \psi_2$ are satisfied if they are satisfied in the rest of the path.

Definition 4.9 (Goal)**.** *The* goal *φ^G of formula φ after occurrence of the set of*

formulas G is defined as follows.

$$\begin{aligned}
(\psi_1 \ \mathcal{U} \ \psi_2)^G &= \top && \text{if } \psi_2 \in G \\
&= \bot && \text{if } \neg\psi_1 \in G \text{ and } \psi_2 \notin G \\
&= \psi_1 \ \mathcal{U} \ \psi_2 && \text{otherwise} \\
(\psi_1 \ \mathcal{W} \ \psi_2)^G &= \top && \text{if } \psi_2 \in G \\
&= \bot && \text{if } \neg\psi_1 \in G \text{ and } \psi_2 \notin G \\
&= \psi_1 \ \mathcal{W} \ \psi_2 && \text{otherwise} \\
(\psi_1 \vee \psi_2)^G &= \psi_1^G \vee \psi_2^G \\
(\psi_1 \wedge \psi_2)^G &= \psi_1^G \wedge \psi_2^G \\
\varphi^G &= \top && \text{if } \varphi \in \Phi
\end{aligned}$$

Example 4.4. *The goal of $(\top \ \mathcal{U} \ p) \wedge (q \ \mathcal{W} \ \bot)$ after occurrence of an atomic formula p, written $((\top \ \mathcal{U} \ p) \wedge (q \ \mathcal{W} \ \bot))^{\{p\}}$, is $\top \wedge (q \ \mathcal{W} \ \bot)$ (which is equivalent to $q \ \mathcal{W} \ \bot$). This expresses that in a state where $\neg q$ has been true, and none of the other relevant formulas have been true, $q \ \mathcal{W} \ \bot$ must be true in the rest of the path, in order for $q \ \mathcal{W} \ \bot$ to be true in the entire path.*

We proceed by expressing the concept of 'between' in path formulas. We say that formula φ is true between sets of formulas G an H if φ is true after all of G and before all of H. To do so, we first express 'before' in path formulas. We say that formula φ is true before a set of formulas H, formula φ is true before all of H.

Definition 4.10 (Before)**.** *Given a set of formulas H, a path λ and a position i, we say that i in λ is* before *H if there does not exist $h \in H$ and k with $0 \leq k \leq i$ such that $M, \lambda[k, \infty] \models h$.*

We add a new path quantifier Before, which is defined as follows.

Definition 4.11 (Before)**.** *Let φ be a formula, and let H be a set of formulas. We define* Before *as follows.*

$$\varphi \ \mathsf{Before} \ H ::= \varphi \ \mathcal{W} \ \bigvee_{h \in H} h.$$

The following lemma shows that the Before path quantifier expresses the concept of 'before'.

Lemma 4.1. *We have $M, \lambda \models \varphi$ Before H if and only if for all i in λ before H, we have that $M, \lambda[i, \infty] \models \varphi$.*

Proof. This follows directly from Definitions 4.10 and 4.11. □

Now we can formally define *between*.

Definition 4.12 (*Between*)**.** *Given sets of formulas G and H, a path λ, and a position i, we say that i in λ is between G and H if for every $g \in G$, there exists k with $0 \leq k \leq i$ such that $M, \lambda[k, \infty] \models g$, and i in λ is before H.*

We define Between recursively as follows in path formulas.

Definition 4.13 (Between)**.** *Let φ be a formula, and let G and H be sets of formulas. We define* Between *as follows.*

$$\varphi \text{ Between } \emptyset, H = \varphi \text{ Before } H$$
$$\varphi \text{ Between } G, H = (\bigwedge_{g \in G} (g \rightarrow \varphi \text{ Between } (G \backslash g), H)) \text{ Before } H$$

Example 4.5. *We demonstrate Definition 4.13 on the case where $|G| = |H| = 1$.*

$$\begin{aligned} \varphi \text{ Between } \{g\}, \{h\} &= (g \rightarrow \varphi \text{ Between } \emptyset, \{h\}) \text{ Before } \{h\} \\ &= (g \rightarrow \varphi \text{ Between } \emptyset, \{h\}) \ \mathcal{W} \ h \\ &= (g \rightarrow \varphi \text{ Before } \{h\}) \ \mathcal{W} \ h \\ &= (g \rightarrow \varphi \ \mathcal{W} \ h) \ \mathcal{W} \ h. \end{aligned}$$

The following lemma shows that the Between path quantifier expresses the concept of 'between'.

Lemma 4.2. *We have $M, \lambda \models \varphi$ Between G, H if and only if for all i in λ between G and H, we have $M, \lambda[i, \infty] \models \varphi$.*

Proof. ($\Rightarrow$) We apply induction on the size of G. In the base case, we have $G = \emptyset$. Assume $M, \lambda \models \varphi$ Between G, H. By Definition 4.13, we have $M, \lambda \models \varphi$ Before H. By Lemma 4.1, we have for all i in λ before H that $M, \lambda[i, \infty] \models \varphi$. Therefore, for all i we have that if for every $g \in G$ there exists k with $0 \leq k \leq i$ such that $M, \lambda[k, \infty] \models g$ and i in λ is before H, then we have $M, \lambda[i, \infty] \models \varphi$. Therefore, by Definition 4.12, we have that for all i in λ between G and H, we have $M, \lambda[i, \infty] \models \varphi$.

We proceed with the induction step. Assume $M, \lambda \models \varphi$ Between G, H. By Definition 4.13, we have $M, \lambda \models (\bigwedge_{g \in G}(g \rightarrow \varphi$ Between $(G \backslash g), H))$ Before H. By Lemma 4.1, we have that for all i in λ before H, we have $M, \lambda[i, \infty] \models (\bigwedge_{g \in G}(g \rightarrow \varphi$ Between $(G \backslash g), H))$. First we assume that there does not exist j such that $\lambda[j, \infty] \models g$ for some $g \in G$ and j in λ is before H. Then, given that $G \neq \emptyset$, there does not exist j such that for every $g \in G$, there exists k with $0 \leq k \leq j$ such that $M, \lambda[k, \infty] \models g$ and j in λ before H. This implies that for all j, if for every $g \in G$ there exists k with $0 \leq k \leq j$ such that $M, \lambda[k, \infty] \models g$ and j in λ before H, then $M, \lambda[i, \infty] \models \varphi$. Now we assume that there does exist j such that $\lambda[j, \infty] \models g$ for some $g \in G$ and j in λ is before H. Let j be the smallest such number, and let g be such that $\lambda[j, \infty] \models g$. Then $M, \lambda[j, \infty] \models (\bigwedge_{g \in G}(g \rightarrow \varphi$ Between $(G \backslash g), H))$. This implies $M, \lambda[j, \infty] \models \varphi$ Between $(G \backslash g), H$. By the induction hypothesis, for all i in $\lambda[j, \infty]$ between $G \backslash g$ and H, we have $M, \lambda[j, \infty][i, \infty] \models \varphi$. Therefore, for all i we have that if for every $g' \in G \backslash g$ there exists k with $0 \leq k \leq i$ such that $M, \lambda[j, \infty][k, \infty] \models g'$ and i in $\lambda[j, \infty]$ is before H, then we have $M, \lambda[i, \infty] \models \varphi$. Since $\lambda[j, \infty] \models g$ and since the existence of k with $0 \leq k \leq j$ such that $\lambda[j, \infty][k, \infty] \models g$ implies the existence of k with $0 \leq k \leq j$ such that $\lambda[k, \infty] \models g$, we have for all j that if for every $g \in G$ there exists k with $0 \leq k \leq j$ such that $M, \lambda[k, \infty] \models g$ and j in λ is before H, then $M, \lambda[j, \infty] \models \varphi$. Therefore, by Definition 4.12, we have for all j in λ between G and H that $M, \lambda[j, \infty] \models \varphi$.

($\Leftarrow$) We apply induction on the size of G. In the base case, we have $G = \emptyset$. Assume that for all i in λ between G and H, we have $M, \lambda[i, \infty] \models \varphi$. Then, by

Definition 4.12, we have for all i that if for every $g \in G$ there exists k with $0 \leq k \leq i$ such that $M, \lambda[k, \infty] \models g$ and i in λ is before H, then we have $M, \lambda[i, \infty] \models \varphi$. Therefore, for all i in λ before H, we have that $M, \lambda[i, \infty] \models \varphi$. This implies by Lemma 4.1 that $M, \lambda \models \varphi$ Before H. Therefore, by Definition 4.13, we have $M, \lambda \models \varphi$ Between G, H.

We proceed with the induction step. Assume that for all i in λ between G and H we have $M, \lambda[i, \infty] \models \varphi$. Therefore, by Definition 4.12, we have for all i that if for every $g \in G$ there exists k with $0 \leq k \leq i$ such that $M, \lambda[k, \infty] \models g$, and i in λ is before H, then $M, \lambda[i, \infty] \models \varphi$. Let i in λ before H, let $g \in G$, and assume that $M, \lambda[i, \infty] \models g$. Let j in $\lambda[i, \infty]$ between $G \backslash g$ and H. Then by Definition 4.12, we have that if for every $g' \in G \backslash g$ there exists k with $0 \leq k \leq j$ such that $M, \lambda[i, \infty][k, \infty] \models g$, and j in $\lambda[i, \infty]$ is before H, then $M, \lambda[i, \infty][j, \infty] \models \varphi$. Let $g' \in G$. If $g = g'$, then $M, \lambda[i, \infty] \models g$, so there exists k with $0 \leq k \leq i + j$ such that $M, \lambda[i + j, \infty] \models g$. If $g \neq g'$, then $g' \in G \backslash g$, so there exists k with $0 \leq k \leq j$ such that $M, \lambda[i, \infty][j, \infty] \models g$, so there exists k with $0 \leq k \leq i + j$ such that $M, \lambda[i, \infty] \models g$. Since j is in $\lambda[i, \infty]$ before H and i is in λ before H, we have that $i + j$ is in λ before H. Therefore, $M, \lambda[i + j, \infty] \models \varphi$, so $M, \lambda[i, \infty][j, \infty] \models \varphi$. This implies that for all i in λ between $G \backslash g$ and H, we have $M, \lambda[i, \infty][j, \infty] \models \varphi$. By the induction hypothesis, we have $\lambda[i, \infty] \models \varphi$ Between $G \backslash g, H$. Therefore, we have $\lambda[i, \infty] \models g \rightarrow (\varphi$ Between $G \backslash g, H)$. This implies $\lambda[i, \infty] \models \bigwedge_{g \in G}(g \rightarrow (\varphi$ Between $G \backslash g, H))$. Therefore, we have that for all i in λ before H, we have $M, \lambda[i, \infty] \models (\bigwedge_{g \in G}(g \rightarrow \varphi$ Between $(G \backslash g), H))$. By Lemma 4.1, we have $M, \lambda \models (\bigwedge_{g \in G}(g \rightarrow \varphi$ Between $(G \backslash g), H))$ Before H. By Definition 4.13, this gives us $M, \lambda \models \varphi$ Between G, H. □

The following lemma shows that given a path λ resulting from a strategy with which a can guarantee φ and a point in that path between $G \subseteq \mathsf{Relv}(\varphi)$ and $\mathsf{Relv}(\varphi) \backslash G$, a can achieve φ^G.

Lemma 4.3. *For all concurrent game structures M,* ATL*-*formulas φ, agents a, $G \subseteq \mathsf{Relv}(\varphi)$, $\lambda \in out(q, \sigma_a)$ such that for all $\lambda' \in out(q, \sigma_a)$ we have $\lambda' \models \varphi_a$, and i in λ between G and $\mathsf{Relv}(\varphi) \backslash G$, we have that $M, \lambda[i] \models \langle\langle a \rangle\rangle \varphi^G$.*

Proof. Let M be a concurrent game structure, q be a state in M, φ be an ATL*-formula, a be an agent, $G \subseteq \mathsf{Relv}(\varphi)$, $\lambda \in out(q, \sigma_a)$ such that for all $\lambda' \in out(q, \sigma_a)$ we have $\lambda' \models \varphi$, and $i \in \mathbb{N}$ between G and $(\mathsf{Relv}(\varphi) \backslash G)$. We will show that $M, \lambda[i] \models \langle\langle a \rangle\rangle \varphi^G$ by structural induction on φ. If $\varphi \in \Phi$ or $\neg\varphi \in \Phi$, then $\varphi^G = \top$, so this holds trivially. Now assume $\varphi = \varphi_1 \wedge \varphi_2$. Then $M, \lambda \models \varphi_1 \wedge \varphi_2$, so $M, \lambda \models \varphi_1$ and $M, \lambda \models \varphi_2$. By induction hypothesis, $M, \lambda[i] \models \langle\langle a \rangle\rangle {\varphi_1}^G$ and $M, \lambda[i] \models \langle\langle a \rangle\rangle {\varphi_2}^G$, so $M, \lambda[i] \models \langle\langle a \rangle\rangle {\varphi_1}^G \wedge \langle\langle a \rangle\rangle {\varphi_2}^G$, so $M, \lambda[i] \models \langle\langle a \rangle\rangle (\varphi_1 \wedge \langle\langle a \rangle\rangle \varphi_2)^G$. The same reasoning holds when $\varphi = \varphi_1 \vee \varphi_2$. Now assume $\varphi = \varphi_1 \,\mathcal{U}\, \varphi_2$. If $\varphi_2 \in G$, then $\varphi^G = \top$, so the claim holds trivially. If $\neg\varphi_1 \in G$ and $\varphi_2 \notin G$, then $M, \lambda \not\models \varphi$, which is a contradiction. Otherwise, $\neg\varphi_1 \notin G$ and $\varphi_2 \notin G$. Let $\lambda^i \in out(\lambda[i], \sigma_a)$, As $\lambda[0, i] \cdot \lambda^i[0, \infty] \in out(q, \sigma_a)$, we have that $M, \lambda[0, i] \cdot \lambda^i[0, \infty] \models \varphi$. Since $\neg\varphi_1 \notin G$, $\varphi_2 \notin G$, and i in λ between G and $\mathsf{Relv}(\varphi) \backslash G$, it holds that $M, \lambda^i[0, \infty] \models \varphi$. This implies that $M, \lambda[i] \models \langle\langle a \rangle\rangle \varphi$. The same reasoning holds when $\varphi = \varphi_1 \,\mathcal{W}\, \varphi_2$. □

Now we give a function τ that translates SSL$^{+\oslash}$-formulas into equivalent ATL*-formulas.

Definition 4.14 (Translation from $\mathrm{SSL}^{+\oslash}$ to ATL^*)**.** *We define function* $\tau : \mathrm{SSL} \to \mathrm{ATL}^*$ *as follows. We set* $\tau(p) = p$ *for* $p \in \Pi$, $\tau(\varphi_1 \vee \varphi_2) = \tau(\varphi_1) \vee \tau(\varphi_2)$ *and* $\tau(\varphi_1 \wedge \varphi_2) = \tau(\varphi_1) \wedge \tau(\varphi_2)$*. We define* $\tau(\exists\ \varphi_a, \exists\ \varphi_b, \varphi_c)$ *as follows:*

$$\begin{aligned} &\langle\!\langle a, b \rangle\!\rangle(\varphi_a \wedge \varphi_b \wedge \varphi_c \\ \wedge &\bigwedge_{G \subseteq \mathsf{Relv}(\varphi_a)} (\langle\!\langle a \rangle\!\rangle {\varphi_a}^G) \text{ Between } G, (\mathsf{Relv}(\varphi_a) \backslash G) \\ \wedge &\bigwedge_{G \subseteq \mathsf{Relv}(\varphi_b)} (\langle\!\langle b \rangle\!\rangle {\varphi_b}^G) \text{ Between } G, (\mathsf{Relv}(\varphi_b) \backslash G)) \end{aligned}$$

Furthermore, we set $\tau(p) = p$ *for* $p \in \Pi$, $\tau(\neg\psi) = \neg\tau(\psi)$*, and* $\tau(\psi_1 \wedge \psi_2) = \tau(\psi_1) \wedge \tau(\psi_2)$*.*

Example 4.6. *Consider the following* SSL*-formula.*

$$\varphi = (\exists\ ((\top\ \mathcal{U}\ p) \wedge (\top\ \mathcal{U}\ q)) \vee (r\ \mathcal{W}\ \top), \exists\ \top, \psi)$$

Then we have

$$\begin{aligned} \tau(\varphi) = \langle\!\langle a, b \rangle\!\rangle((((\top\ \mathcal{U}\ p) \wedge (\top\ \mathcal{U}\ q)) \vee (r\ \mathcal{W}\ \top)) \wedge \top \wedge \psi \wedge \\ \langle\!\langle a \rangle\!\rangle((\top\ \mathcal{U}\ p) \wedge (\top\ \mathcal{U}\ q)\ ) \vee (r\ \mathcal{W}\ \bot) \text{ Between } \emptyset, \{p, q, \neg r\} \\ \wedge \langle\!\langle a \rangle\!\rangle(\top\ \ \wedge (\top\ \mathcal{U}\ q)\ ) \vee (r\ \mathcal{W}\ \bot) \text{ Between } \{p\}, \{q, \neg r\} \\ \wedge \langle\!\langle a \rangle\!\rangle((\top\ \mathcal{U}\ p) \wedge \top\ \ \ ) \vee (r\ \mathcal{W}\ \bot) \text{ Between } \{q\}, \{p, \neg r\} \\ \wedge \langle\!\langle a \rangle\!\rangle(\top\ \ \wedge \top\ \ \ ) \vee (r\ \mathcal{W}\ \bot) \text{ Between } \{p, q\}, \{\neg r\} \\ \wedge \langle\!\langle a \rangle\!\rangle((\top\ \mathcal{U}\ p) \wedge (\top\ \mathcal{U}\ q)\ ) \vee \bot \text{ Between } \{\neg r\}, \{p, q\} \\ \wedge \langle\!\langle a \rangle\!\rangle(\top\ \ \wedge (\top\ \mathcal{U}\ q)\ ) \vee \bot \text{ Between } \{p, \neg r\}, \{q\} \\ \wedge \langle\!\langle a \rangle\!\rangle((\top\ \mathcal{U}\ p) \wedge \top\ \ \ ) \vee \bot \text{ Between } \{q, \neg r\}, \{p\} \\ \wedge \langle\!\langle a \rangle\!\rangle(\top\ \ \wedge \top\ \ \ ) \vee \bot \text{ Between } \{p, q, \neg r\}, \emptyset). \end{aligned}$$

For example, the subformula

$$\langle\!\langle a \rangle\!\rangle(\top \wedge \top) \vee (r\ \mathcal{W}\ \bot) \text{ Between } \{p, q\}, \{\neg r\}$$

expresses that in all states in a path where p *and* q *have been true before, and* $\neg r$ *has not been true,* a *should be able to guarantee either* $\top$ *or* $r\ \mathcal{W}\ \bot$*.*

4.3.4 Correctness of the Translation

Now we show that translation τ is correct, i.e., that in any model, φ is true if and only if $\tau(\varphi)$ is true.

Theorem 4.1. *For all turn-based game structures* M*, states* q *in* M*, and formulas* $\varphi \in \mathrm{SSL}$*, it holds that* $M, q \models \varphi$ *if and only if* $M, q \models \tau(\varphi)$*.*

Proof. We prove this by induction on the structure of φ. If $\varphi \in \Pi$, $\varphi = \varphi_1 \wedge \varphi_2$ or $\varphi = \neg\varphi'$, the result follows directly from the induction hypothesis. We proceed with the case where $\varphi = (\exists\ \varphi_a, \exists\ \varphi_b, \varphi_c)$. ($\Rightarrow$) Assume that $M, q \models \varphi$. Let $\sigma_a \in \Sigma_a$ and $\sigma_b \in \Sigma_b$ such that for all $\lambda \in \mathit{out}(q, (\sigma_a, \sigma_b))$, we have $M, \lambda \models \varphi_c$; for all $\sigma'_b \in \Sigma_b$ and $\lambda^a \in \mathit{out}(q, (\sigma_a, \sigma'_b))$, we have $M, \lambda^a \models \varphi_a$; and for all $\sigma'_a \in \Sigma_a$ and $\lambda^b \in \mathit{out}(q, (\sigma'_a, \sigma_b))$, we have $M, \lambda^b \models \varphi_b$.

Let $\lambda \in out(q, (\sigma_a, \sigma_b))$. By definition of σ_a and σ_b, we have $M, \lambda \models \varphi_a$, $M, \lambda \models \varphi_b$, and $M, \lambda \models \varphi_c$. Since $\lambda \in out(q, (\sigma_a, \sigma_b))$, we have that $\lambda \in out(q, (\sigma_a))$. Moreover, for all $\lambda' \in out(q, \sigma_a)$, we have $\lambda' \models \varphi_a$. Now let $G \subseteq \mathsf{Relv}(\varphi)$, and i in λ between G and $\mathsf{Relv}(\varphi_a)\backslash G$. Then we have $M, \lambda[i] \models \langle\langle a \rangle\rangle {\varphi_a}^G$ by Lemma 4.3. This implies that $M, \lambda \models \bigwedge_{G \subseteq \mathsf{Relv}(\varphi_a)} (\langle\langle a \rangle\rangle {\varphi_a}^G)$ Between $G, (\mathsf{Relv}(\varphi_a)\backslash G)$ by Lemma 4.2.

In the same way, we prove $M, \lambda \models \bigwedge_{G \subseteq \mathsf{Relv}(\varphi_b)} (\langle\langle b \rangle\rangle {\varphi_b}^G)$ Between $G, (\mathsf{Relv}(\varphi_b)\backslash G)$. Therefore, $M, q \models \tau(\varphi)$.

($\Leftarrow$) Assume that $M, q \models \tau(\varphi)$. Let $\sigma_a \in \Sigma_a$ and $\sigma_b \in \Sigma_b$ be such that for all $\lambda \in out(q, \sigma_a, \sigma_b)$, we have $M, \lambda \models \tau(\varphi)$.

Let $\lambda \in out(q, (\sigma_a, \sigma_b))$. Then we have $M, \lambda \models \varphi_a$, $M, \lambda \models \varphi_b$, and $M, \lambda \models \varphi_c$ by definition of σ_a and σ_b.

For every $i \in \mathbb{N}$, let G^i be such that i in λ between G^i and $\mathsf{Relv}(\varphi_a)\backslash G^i$. Now let $\sigma_a^i \in \Sigma_a$ be for every $i \in \mathbb{N}$ a strategy such that for every $\lambda^a \in out(\lambda[i], \sigma_a^i)$, we have $M, \lambda^a \models \varphi_a^{G^i}$.

We construct a strategy σ_a^* as follows:

- If λ' is a prefix of λ, then $\sigma_a^*(\lambda') = \sigma_a(\lambda')$.
- Otherwise, let i be the largest integer such that $\lambda'[0, i]$ is a prefix of λ and let $\sigma_a^*(\lambda') = \sigma_a^i(\lambda')$.

Now we show that for all $\lambda^* \in out(q, \sigma_a^*)$, we have $M, \lambda^* \models \varphi_a$. Let $\lambda^* \in out(q, \sigma_a^*)$. If $\lambda^* = \lambda$, then $M, \lambda^* \models \varphi_a$. Otherwise, let i be the largest integer such that $\lambda^*[0, i]$ is a prefix of λ. Let $G \subseteq \mathsf{Relv}(\varphi_a)$ be such that i between G and $\mathsf{Relv}(\varphi_a)\backslash G$.

If $\lambda[i]$ is an a-state, then $\lambda[i+1]$ is also a prefix of λ, which is a contradiction. Therefore, i is a b-state.

Since i is a b-state and i is the largest integer such that $\lambda^*[0, i]$ is a prefix of λ, we have that $out(\lambda[i], \sigma_a^i) = out(\lambda[i], \sigma_a^*)$. We know that for all $\lambda' \in out(\lambda[i], \sigma_a^i)$, it holds that $M, \lambda' \models \varphi^G$, we also have that for all $M, \lambda' \in out(\lambda[i], \sigma_a^*)$, it holds that $M, \lambda' \models \varphi^G$. Therefore, $M, \lambda^*[i] \models {\varphi_a}^G$.

Now we prove by induction on φ_a that if $M, \lambda^*[i, \infty] \models \varphi_a^G$ and $M, \lambda \models \varphi_a$, then $M, \lambda^* \models \varphi_a$. If $\varphi \in \Pi$ or $\varphi \in \Phi$, then $\varphi^G = \top$, so this holds trivially. If $\varphi = \varphi_1 \wedge \varphi_2$, then $M, \lambda^*[i, \infty] \models (\varphi_1 \wedge \varphi_2)^G$, so $M, \lambda^*[i, \infty] \models \varphi_1^G \wedge \varphi_2^G$ by definition of G. Therefore, by induction hypothesis, $M, \lambda^* \models \varphi_1$ and $M, \lambda^* \models \varphi_2$, so $M, \lambda^* \models \varphi_1 \wedge \varphi_2$. The proof for $\varphi = \varphi_1 \vee \varphi_2$ is similar. Now assume $\varphi = \varphi_1 \,\mathcal{U}\, \varphi_2$. If $\varphi_2 \in G$ then this holds trivially. If $\neg\varphi_1 \in G$ and $\varphi_2 \notin G$ then $M, \lambda^* \not\models \varphi_a$, which is a contradiction. Otherwise, $\neg\varphi_1 \notin G$ and $\varphi_2 \notin G$. By $\neg\varphi_1 \notin G$, we have that for all $j \leq i$, $M, \lambda^*[j, \infty] \not\models \varphi_1$. Since $(\varphi_1 \,\mathcal{U}\, \varphi_2)^G = \varphi_1 \,\mathcal{U}\, \varphi_2$, there exists $j \geq i$ such that $M, \lambda^*[j, \infty] \models \varphi_2$ and for all $k < j$, $M, \lambda^*[k, \infty] \models \varphi_1$. This implies that there exists $j \geq 0$ such that $M, \lambda^*[j, \infty] \models \varphi_2$ and for all $k < j$, $M, \lambda^*[k, \infty] \models \varphi_1$, which implies that $M, \lambda^* \models \varphi_1 \,\mathcal{U}\, \varphi_2$. The proof for $\varphi = \varphi_1 \,\mathcal{W}\, \varphi_2$ is similar.

This shows that for all $\lambda^* \in out(q, \sigma_a^*)$, we have $M, \lambda^* \models \varphi_a$.

Symmetrically, we can prove that for all $\lambda^* \in out(q, \sigma_b^*)$, we have $M, \lambda^* \models \varphi_b$.

Let $\lambda' \in out(q, (\sigma_a^*, \sigma_b^*))$. We prove by induction on i that $\lambda'[i] = \lambda[i]$. We have that $\lambda[0] = q = \lambda'[0]$. Assume the claim holds for i. By induction hypothesis, $\lambda'[0, i]$

is a prefix of λ, so $sigma_a^*(\lambda[i]) = sigma_a(\lambda[i])$ and $sigma_b^*(\lambda[i]) = sigma_b(\lambda[i])$. This implies that $\lambda'[i+1] = \lambda[i+1]$. Therefore, $\lambda' = \lambda$. As $M, \lambda \models \varphi_c$, this implies that for all $\lambda^* \in out(q, (\sigma_a^*, \sigma_b^*))$, we have $M, \lambda^* \models \varphi_c$.

This proves that $M, q \models (\exists\, \varphi_a, \exists\, \varphi_b, \varphi_c)$. □

From Theorem 4.1, it follows directly that ATL* is at least as expressive as SSL$^{+\oslash}$ on turn-based game structures.

Corollary 4.1. *If $\mathcal{M}$ is the class of turn-based game structures, we have* SSL$^{+\oslash} \sqsubseteq_{\mathcal{M}}$ ATL*.

4.4 Expressive Power in Concurrent Game Structures

We proceed by looking at concurrent game structures. We show that in concurrent game structures, SSL$^{+\oslash}$ is in fact more expressive than ATL*. In particular, we show that in concurrent game structures, only some trivial SSL-formulas can be expressed in ATL*. We prove this by showing that for non-trivial ATL*-formulas, we can provide two models that can be distinguished by a SSL-formula, but not by an ATL*-formula. To prove that the models cannot be distinguished by an ATL*-formula, we extend the notion of bisimulation for ATL [ÅGJ07, DJ10] to ATL* with imperfect information.

4.4.1 ATL*-bisimulation for Perfect Information

Definition 4.15 (ATL*-bisimulation)**.** *Let $M_1 = (\mathbb{A}\mathrm{gt}, Act_1, Q_1, \Pi, \pi_1, d_1, \delta_1)$ and $M_2 = (\mathbb{A}\mathrm{gt}, Act_2, Q_2, \Pi, \pi_2, d_2, \delta_2)$ be concurrent game structures, let $C \subseteq \mathbb{A}\mathrm{gt}$ be a set of agents, and let Σ_{C1} and Σ_{C2} be the set of joint strategies of agents C in M_1 and M_2, respectively. We say that a relation $\beta \subseteq Q_1 \times Q_2$ is a C-ATL*-bisimulation between M_1 and M_2, denoted $M_1 \rightleftarrows_\beta^C M_2$, if for all $q_1 \in Q_1$ and $q_2 \in Q_2$, $q_1\ \beta\ q_2$ implies that*

Local harmony *$\pi_1(q_1) = \pi_2(q_2)$.*

Forth *For every joint strategy $\sigma_{C1} \in \Sigma_C$ in M_1, there exists a joint strategy $\sigma_{C2} \in \Sigma_{C2}$ such that for every path $q_2^1, q_2^2, \ldots \in out(q_2, \sigma_{C2})$, there exists a path $q_1^1, q_1^2, \ldots \in out(q_1, \sigma_{C1})$ such that $q_2^i\ \beta\ q_2^i$ for all $i > 0$.*

Back *Likewise, for 1 and 2 swapped.*

If β is a C-ATL-bisimulation between M_1 and M_2 for every $C \subseteq \mathbb{A}\mathrm{gt}$, we call it a* ATL*-bisimulation between M_1 and M_2, denoted $M_1 \rightleftarrows_\beta M_2$.*

If $M_1 \rightleftarrows_\beta M_2$ and $q_1\ \beta\ q_2$, then we also say that β is a local ATL*-bisimulation between (M_1, q_1) and (M_2, q_2), denoted $(M_1, q_1) \rightleftarrows_\beta (M_2, q_2)$.*

We say that two paths $\lambda_1 = q_1^0, q_1^1, \ldots$ and $\lambda_2 = q_2^0, q_2^1, \ldots$ are ATL*-bisimilar, written $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$, when $(M_1, q_1^i) \rightleftarrows_\beta (M_2, q_2^i)$ for all $i \geq 0$.*

Example 4.7. *In Figure 4.1, we have $M_1 \rightleftarrows_\beta M_2$ with $\beta(q_0) = q_0$.*

The following theorem states that in two states that are bisimilar, the same state

formulas are true, and in two paths that are bisimilar, the same path formulas are true.

Theorem 4.2. *If $M_1 \rightleftarrows_\beta M_2$ then:*

1. *If φ is a state formula and $(M_1, q_1) \rightleftarrows_\beta (M_2, q_2)$, then $M_1, q_1 \models \varphi$ if and only if $M_2, q_2 \models \varphi$.*
2. *If φ is a path formula and $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$, then $M_1, \lambda_1 \models \varphi$ if and only if $M_2, \lambda_2 \models \varphi$.*

Proof. We assume that $(M_1, q_1) \rightleftarrows_\beta (M_2, q_2)$ and $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$. We prove the theorem by induction on the structure of φ. First we prove 1. The base case follows from the fact that we have $\varphi \in \Pi$, so the theorem holds by local harmony. To prove the induction step, observe that the cases for $\vee$ and $\neg$ follow directly from the induction hypothesis. For the case $\langle\langle a \rangle\rangle \varphi$, we assume that $M_1, q_1 \models \langle\langle C \rangle\rangle \psi$, so there exists a joint strategy σ_C such that for all paths $\lambda \in out(q_1, \sigma_{C1})$, we have $M_1, \lambda \models \psi$. Let $\sigma_{C1} \in \Sigma_{C1}$. Then by *forth*, there exists a joint strategy $\sigma_{C2} \in \Sigma_{C2}$ such that for every $\lambda_2 \in out(q_2, \sigma_{C2})$, there exists $q_1' \sim_c q_1$ and $\lambda_1 \in out(q_1', \sigma_{C1})$ such that $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$. Now let σ_{C2} be such a strategy and $\lambda_2 \in out(q_2, \sigma_{C2})$. Then there exists $\lambda_1 \in out(q_1, \sigma_{C1})$ such that $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$, so by induction hypothesis, we have that $M_2, \lambda_2 \models \varphi$. This implies that there exists a joint strategy ${\sigma_2}^C$ such that for all and paths $\lambda_2 \in out(q_2, \sigma_c)$, we have $M_2, \lambda_2 \models \varphi_2$. We proceed with proving 2. The cases for $\vee$ and $\neg$ follow directly from the induction hypothesis. The cases for $\bigcirc$, $\Box$ and $\mathcal{U}$ follow from the fact that $(M_1, \lambda_1) \rightleftarrows_\beta^C (M_2, \lambda_2)$ and the induction hypothesis. The 'if' direction follows by symmetry. □

4.4.2 Results

Definition 4.16 (φ_1, φ_2, and φ_3)**.** *Let φ_1, φ_2, and φ_3 be defined as follows.*

1. $\varphi_1 = \neg(\varphi_a \wedge \varphi_b \wedge \varphi_c)$;
2. $\varphi_2 = \neg(\varphi_a \wedge \varphi_b \wedge \neg\varphi_c)$;
3. $\varphi_3 = \varphi_a \wedge \varphi_b$.

We show that an SSL-formula only has a corresponding ATL*-formula if in every model, at least one of φ_1, φ_2, and φ_3 is true in every path.

Theorem 4.3. *If $\models \langle\langle\rangle\rangle \varphi_1 \vee \langle\langle\rangle\rangle \varphi_2 \vee \langle\langle\rangle\rangle \varphi_3$, then*

$$[\![(\exists\, \varphi_a, \exists\, \varphi_b, \varphi_c)]\!] = (\langle\langle\rangle\rangle \varphi_1 \rightarrow \bot) \wedge (\langle\langle\rangle\rangle \varphi_2 \rightarrow \langle\langle a \rangle\rangle \varphi_a \wedge \langle\langle b \rangle\rangle \varphi_b) \wedge (\langle\langle\rangle\rangle \varphi_3 \rightarrow \langle\langle a, b \rangle\rangle \varphi_c$$

where $[\![\cdot]\!]$ is a mapping from SSL*-formulas to* ATL* *formulas such that $M, q \models [\![\varphi]\!]$ if and only if $M, q \models \varphi$ for every* SSL*-formula φ, concurrent game structure M, and state q in M.*

Proof. Assume that $\models \langle\langle\rangle\rangle\varphi_1 \vee \langle\langle\rangle\rangle\varphi_2 \vee \langle\langle\rangle\rangle\varphi_3$. Let q be a state in concurrent game structure M. First we show that if $M, q \models \psi$ where $\psi = (\exists\ \varphi_a, \exists\ \varphi_b, \varphi_c)$, then $M, q \models [\![\psi]\!]$. We assume that $M, q \models \psi$, and proceed by proving that the three conjuncts of $[\![\psi]\!]$ hold.

- Assume that $M, q \models \langle\langle\rangle\rangle\varphi_1$. Then for all paths $\lambda \in out(q, \emptyset)$, we have $M, \lambda \models \neg(\varphi_a \wedge \varphi_b \wedge \varphi_c)$. Furthermore, by $M, q \models \psi$, there exists $\sigma_a \in \Sigma_a$ and $\sigma_b \in \Sigma_b$ such that for all $\lambda \in out(q, (\sigma_a, \sigma_b))$, we have $M, \lambda \models \varphi_a \wedge \varphi_b \wedge \varphi_c$. This is a contradiction, so $M, q \models \bot$, and thus $M, q \models \langle\langle\rangle\rangle\varphi_1 \rightarrow \bot$.
- By definition of ψ, we have $M, q \models \langle\langle a\rangle\rangle\varphi_a \wedge \langle\langle b\rangle\rangle\varphi_b$. This implies that $M, q \models \langle\langle\rangle\rangle\varphi_2 \rightarrow \langle\langle a\rangle\rangle\varphi_a \wedge \langle\langle b\rangle\rangle\varphi_b$.
- By definition of ψ, we have $M, q \models \langle\langle a, b\rangle\rangle\varphi_c$. This implies that $M, q \models \langle\langle\rangle\rangle\varphi_3 \rightarrow \langle\langle a, b\rangle\rangle\varphi_c$.

Now we show that if $M, q \models [\![\psi]\!]$, then $M, q \models \psi$. We prove this by case distinction on the disjuncts of $\langle\langle\rangle\rangle\varphi_1 \vee \langle\langle\rangle\rangle\varphi_2 \vee \langle\langle\rangle\rangle\varphi_3$.

- Assume that $M, q \models \langle\langle\rangle\rangle\varphi_1$. Then $M, q \models \bot$ by $M, q \models [\![\psi]\!]$, so $M, q \models \psi$.
- Assume that $M, q \models \langle\langle\rangle\rangle\varphi_2$. Then for all paths $\lambda \in out(q, \emptyset)$, we have $M, \lambda \models \neg(\varphi_a \wedge \varphi_b \wedge \neg\varphi_c)$ and thus $M, \lambda \models (\varphi_a \wedge \varphi_b) \rightarrow \varphi_c$. Moreover, we have $M, q \models \langle\langle a\rangle\rangle\varphi_a \wedge \langle\langle b\rangle\rangle\varphi_b$ by definition of ψ. Then there exists $\sigma_a \in \Sigma_a$, $\sigma_b \in \Sigma_b$ such that for all $\sigma'_b \in \Sigma_b$ and $\lambda \in out(q, (\sigma_a, \sigma'_b))$, we have $M, \lambda \models \varphi_a$, and for all $\sigma'_a \in \Sigma_a$ and $\lambda \in out(q, (\sigma'_a, \sigma_b))$, we have $M, \lambda \models \varphi_b$. Therefore, for all $\lambda \in out(q, (\sigma_a, \sigma_b))$, we have $M, \lambda \models \varphi_a$ and $M, \lambda \models \varphi_b$, and thus $M, \lambda \models \varphi_c$. This shows that $M, q \models \psi$.
- Assume that $M, q \models \langle\langle\rangle\rangle\varphi_3$. Then for all paths $\lambda \in out(q, \emptyset)$, we have $M, \lambda \models \varphi_a \wedge \varphi_b$. Moreover, we have $M, q \models \langle\langle a, b\rangle\rangle\varphi_c$ by $M, q \models [\![\psi]\!]$, Then there exists $\sigma_a \in \Sigma_a$, $\sigma_b \in \Sigma_b$ such that for all $\lambda \in out(q, (\sigma_a, \sigma_b))$, we have $M, \lambda \models \varphi_c$. Moreover, for all $\sigma'_b \in \Sigma_b$ and $\lambda \in out(q, (\sigma_a, \sigma'_b))$, we have $M, \lambda \models \varphi_a$, and for all $\sigma'_a \in \Sigma_a$ and $\lambda \in out(q, (\sigma'_a, \sigma_b))$, we have $M, \lambda \models \varphi_b$. Therefore, $M, q \models \psi$.

□

Theorem 4.4. *If M is a concurrent game structure such that $M \not\models \langle\langle\rangle\rangle\varphi_1 \vee \langle\langle\rangle\rangle\varphi_2 \vee \langle\langle\rangle\rangle\varphi_3$, then $\psi = (\exists\ \varphi_a, \exists\ \varphi_b, \varphi_c)$ cannot be expressed in* ATL*.

Proof. Assume that $\not\models \langle\langle\rangle\rangle\neg\varphi_1 \vee \langle\langle\rangle\rangle\varphi_2 \vee \langle\langle\rangle\rangle\varphi_3$. Then $\neg(\langle\langle\rangle\rangle\neg\varphi_1 \vee \langle\langle\rangle\rangle\varphi_2 \vee \langle\langle\rangle\rangle\varphi_3) = \exists\varphi_1 \wedge \exists\varphi_2 \wedge \exists\varphi_3$ is satisfiable. Without loss of generality, we assume that $\exists\varphi_1 \wedge \exists\varphi_2 \wedge \exists\neg\varphi_a$ is satisfiable. We turn models M_1 and M_2 in 4.1 into an imperfect-information concurrent game by the standard construction for turning concurrent games into imperfect-information turn-based games. One can easily check that the models thus obtained are ATL*-bisimilar. Furthermore, $M_1, q_0 \not\models \psi$, while $M_2, q_0 \models \psi$. This shows that ψ cannot be expressed in ATL. □

Corollary 4.2. *If M is a concurrent game structure, $\psi = (\exists\ \varphi_a, \exists\ \varphi_b, \varphi_c)$ can be expressed in* ATL* *if and only if $\models \langle\langle\rangle\rangle\neg\varphi_a \vee \langle\langle\rangle\rangle\varphi_b \vee \langle\langle\rangle\rangle\varphi_c$.*

Proof. This follows directly from Theorem 4.3 and Theorem 4.4. □

Corollary 4.3. *If $\mathcal{M}$ is the set of imperfect-information concurrent game structures,* SSL $\not\sqsubseteq_{\mathcal{M}}$ ATL*.

Proof. This follows directly from Theorem 4.4. □

4.5 Expressive Power in Imperfect-information Models

Just like for concurrent game structures, we prove that SSL$^{+\oslash}$, and therefore also SSL, is more expressive than ATL*, by providing two models that can be distinguished by a SSL$^{+\oslash}$-formula, but not by an ATL*-formula. To prove that the models cannot be distinguished by an ATL*-formula, we extend ATL*-bisimulation to imperfect-information concurrent game structures.

Our definition differs from traditional definitions of bisimulation in that we do not define ATL*-bisimulation in terms of the possible next states from a state, but in terms of the possible paths from a state.

Definition 4.17 (ATL*-bisimulation)**.** *Let $M_1 = ((\mathbb{A}\mathrm{gt}, Act_1, Q_1, \Pi, \pi_1, d_1, \delta_1), \sim_1)$ and $M_2 = ((\mathbb{A}\mathrm{gt}, Act_2, Q_2, \Pi, \pi_2, d_2, \delta_2), \sim_2)$ be concurrent game structures, let $C \subseteq \mathbb{A}\mathrm{gt}$ be a set of agents, and let Σ_{C1} and Σ_{C2} be the set of joint strategies of agents C in M_1 and M_2, respectively. We say that $\beta \subseteq Q_1 \times Q_2$ is a C-ATL*-bisimulation between M_1 and M_2, denoted $M_1 \rightleftarrows^C_\beta M_2$, if for all $q_1 \in Q_1$ and $q_2 \in Q_2$, $q_1\ \beta\ q_2$ implies that*

Local harmony *$\pi_1(q_1) = \pi_2(q_2)$;*

Forth *For every joint strategy $\sigma_{C1} \in \sigma_{C1}$, there exists a joint strategy $\sigma_{C2} \in \sigma_{C2}$ such that for every agent $a \in C$ with $q'_2 \sim_a q_2$ and for every path $q^1_2, q^2_2, \ldots \in out(q'_2, \sigma_{C2})$, there exists an agent $a \in C$ with $q'_1 \sim_a q_1$ and a path $q^1_1, q^2_1, \ldots \in out(q'_1, \sigma_{C1})$ such that $q^i_2\ \beta\ q^i_2$ for all $i > 0$.*

Back *Likewise, for 1 and 2 swapped.*

If β is a C-ATL-bisimulation between M_1 and M_2 for every $C \subseteq \Sigma$, we call it an* ATL*-bisimulation between M_1 and M_2, denoted $M_1 \rightleftarrows_\beta M_2$.*

If $M_1 \rightleftarrows_\beta M_2$ and $q_1\ \beta\ q_2$, then we also say that β is a local ATL*-bisimulation between (M_1, q_1) and (M_2, q_2), denoted $(M_1, q_1) \rightleftarrows_\beta (M_2, q_2)$.*

We say that two paths $\lambda_1 = q^0_1, q^1_1, \ldots$ and $\lambda_2 = q^0_2, q^1_2, \ldots$ are ATL*-bisimilar, written $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$, when $(M_1, q^i_1) \rightleftarrows_\beta (M_2, q^i_2)$ for all $i \geq 0$.*

The following theorem states that if two states are bisimilar, the same state formulas are true in these states, and if two paths are bisimilar, the same path formulas are true in these paths.

Theorem 4.5. *If $M_1 \rightleftarrows_\beta M_2$ then:*

1. *If φ is a state formula and $(M_1, q_1) \rightleftarrows_\beta (M_2, q_2)$, then $M_1, q_1 \models \varphi$ if and only if $M_2, q_2 \models \varphi$.*

2. *If φ is a path formula and $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$, then $M_1, \lambda_1 \models \varphi$ if and only if $M_2, \lambda_2 \models \varphi$.*

Proof. We assume that $(M_1, q_1) \rightleftarrows_\beta (M_2, q_2)$ and $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$. We prove the theorem by induction on the structure of φ. First we prove 1. The base case follows from the fact that we have $\varphi \in \Pi$, so the theorem holds by local harmony. To prove the induction step, observe that the cases for $\vee$ and $\neg$ follow directly from the induction hypothesis. For the case $\langle\langle a \rangle\rangle \varphi$, we assume that $M_1, q_1 \models \langle\langle C \rangle\rangle \psi$, so there exists a joint strategy σ_c such that for all agents $a \in C$, states $q_1' \sim_a q_1$ and paths $\lambda \in out(q_1', \sigma_{C1})$, we have $M_1, \lambda \models \psi$. Let $\sigma_{C1} \in \sigma_{C1}$. Then by *forth*, there exists a joint strategy $\sigma_{C2} \in \sigma_{C2}$ such that for every $q_2' \sim_c q_2$ and for every $\lambda_2 \in out(q_2', \sigma_{C2})$, there exists $q_1' \sim_c q_1$ and $\lambda_1 \in out(q_1', \sigma_{C1})$ such that $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$. Now let σ_{C2} be such a strategy and let $q_2' \sim_c q_2$ and $\lambda_2 \in out(q_2', \sigma_{C2})$. Then there exists $q_1' \sim_c q_1$ and $\lambda_1 \in out(q_1', \sigma_{C1})$ such that $(M_1, \lambda_1) \rightleftarrows_\beta (M_2, \lambda_2)$, so by induction hypothesis, we have that $M_2, \lambda_2 \models \varphi$. This implies that there exists a joint strategy ${\sigma_2}^C$ such that for all agents $a \in C$, states $q_2' \sim_a q_2$ and paths $\lambda_2 \in out(q_2', \sigma_c)$, we have $M_2, \lambda_2 \models \varphi_2$. We proceed with proving 2. The cases for $\vee$ and $\neg$ follow directly from the induction hypothesis. The cases for $\bigcirc$, $\Box$ and $\mathcal{U}$ follow from the fact that $(M_1, \lambda_1) \rightleftarrows_\beta^C (M_2, \lambda_2)$ and the induction hypothesis. The 'if' direction follows by symmetry. □

Now we can prove that in imperfect-information concurrent game structures, ATL* is not as expressive as SSL.

Theorem 4.6. *If $\mathcal{M}$ is the set of imperfect-information turn-based game structures,* SSL $\not\sqsubseteq_{\mathcal{M}}$ ATL*.

Proof. One can easily check that M_1 and M_2 in Figure 4.2 are bisimilar. However, $M_1, q_1 \models (\exists\, \varphi_a, \exists\, \varphi_b, \varphi_c)$, while $M_2, q_0 \not\models (\exists\, \varphi_a, \exists\, \varphi_b, \varphi_c)$. □

Corollary 4.4. *If $\mathcal{M}$ is the set of imperfect-information concurrent game structures,* SSL $\not\sqsubseteq_{\mathcal{M}}$ ATL*.

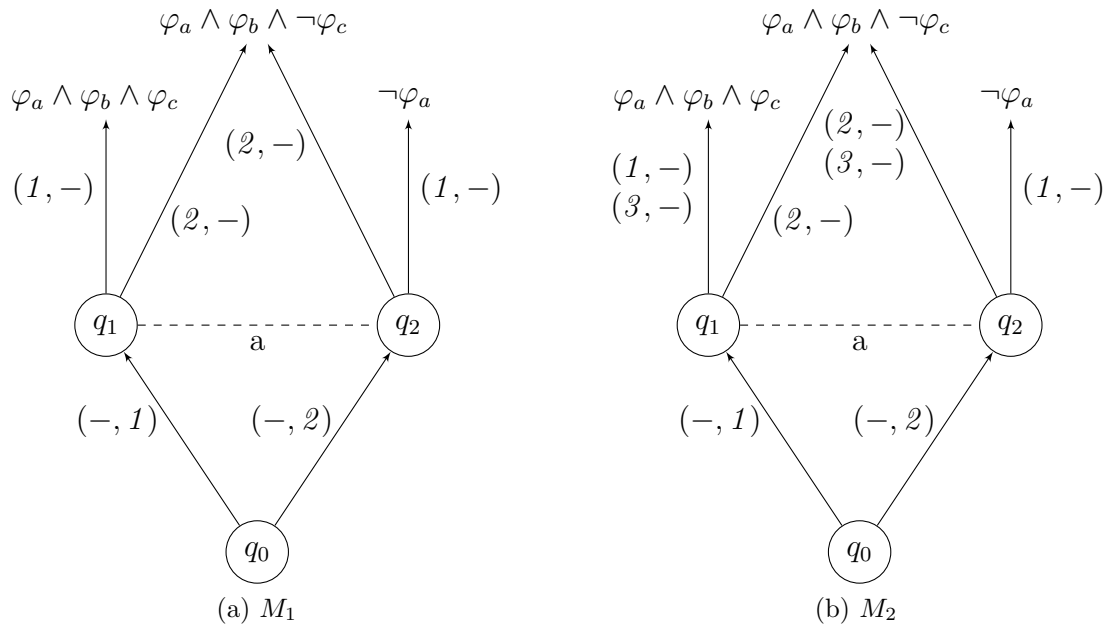
Proof. This follows from the fact that every imperfect-information turn-based game structure is also an imperfect-information concurrent game structure. □

4.6 Application to Security Properties

Winning secure equilibria [CHJ06] express that two agents a and b have a joint strategy to achieve their respective goals ψ_a and ψ_b, which are expressed as path formulas [CHP10]. Moreover, if an agent plays his part of the joint strategy, he can guarantee that if the goal of the other agent gets satisfied, his own goal gets satisfied as well. The concept of winning secure equilibria can be expressed in SSL as follows [CHP10]:

$$(\exists\, \varphi_a \rightarrow \varphi_b, \exists\, \varphi_b \rightarrow \varphi_a, \varphi_a \wedge \varphi_b).$$

In the original paper [CHP10], it was stated that "such a condition is difficult to state without explicit quantification over strategies". From our results, it follows



(a) M_1 (b) M_2

Figure 4.2: Two bisimilar models

that winning secure equilibria in fact can be expressed in ATL*, as long as one is restricted to turn-based perfect-information game structures. On the other hand, our results formally confirm that winning secure equilibria cannot be expressed in concurrent game structures and imperfect-information game structures.

Effective fairness can be seen as an instance of winning secure equilibria. In order for a fair-exchange protocol to be correct, the protocol must satisfy that an agent has a strategy that both guarantees fairness and effectiveness [CR10, JMM12]. Formally, i.e., there exists a pair of strategies (σ_a, σ_b) such that:

1. If Alice plays σ_a, then her goal $\Box(p_a \rightarrow (\Diamond p_b))$ is achieved;
2. If Bob plays σ_b, then his goal $\Box(p_b \rightarrow (\Diamond p_a))$ is achieved;
3. If Alice plays σ_a and Bob plays σ_b, then the common goal $\Diamond(p_a \wedge p_b)$ is achieved.

Therefore, effective fairness can be expressed by the following strategy logic formula:

$$\textsc{EffFair}_{SL} = (\exists\, \Box(p_b \rightarrow (\Diamond p_a)), \exists\, \Box(p_a \rightarrow (\Diamond p_b)), \Diamond(p_a \wedge p_b))$$

Theorem 4.7. $\textsc{EffFair}_{SL}$ *cannot be expressed in* ATL* *in concurrent game structures or imperfect-information game structures.*

Proof. Let $\varphi_a = \Box(p_b \rightarrow (\Diamond p_a))$, $\varphi_b = \Box(p_a \rightarrow (\Diamond p_b))$, and $\varphi_c = \Diamond(p_a \wedge p_b)$. First we show the case for concurrent game structures. Consider concurrent game structures M_1 and M_2 from Figure 4.1. We have $M_1, q_0 \not\models \textsc{EffFair}_{SL}$ and $M_2, q_0 \models \textsc{EffFair}_{SL}$. As M_1 and M_2 are ATL*-bisimilar, $\textsc{EffFair}_{SL}$ cannot be expressed in concurrent game structures in ATL*.

Now consider concurrent game structures M_1 and M_2 from Figure 4.2. We have $M_1, q_0 \not\models \textsc{EffFair}_{SL}$ and $M_2, q_0 \models \textsc{EffFair}_{SL}$. As M_1 and M_2 are ATL*-bisimilar, $\textsc{EffFair}_{SL}$ cannot be expressed in imperfect-information structures in ATL*. □

4.7 Conclusion

We provided a translation from SSL$^{+\oslash}$, the unnested simple one-alternating fragment of Strategy Logic without $\bigcirc$, into ATL*, on turn-based game structures. This shows that SSL$^{+\oslash}$ is at least as expressive as ATL* on such structures. The logic SSL$^{+\oslash}$ is an interesting fragment, because it can express useful properties from game theory and security protocols, such as secure equilibria. Furthermore, we pointed out a problem in the proof of [CR10] that the full one-alternating fragment of Strategy Logic SSL is at least as expressive as ATL* in turn-based game structures. It remains therefore an open question whether ATL* and the full logic SSL are equally expressive.

Conversely, we pointed out that in concurrent game structures, SSL$^{+\oslash}$ is in fact strictly more expressive than ATL*. Moreover, we gave a characterization of the class of SSL formulas that cannot be expressed in ATL* in such game structures. Finally, we proposed an extension of SSL to imperfect-information concurrent game structures, and show that in such structures, SSL$^{+\oslash}$ is also strictly more expressive than ATL*.

Our results have consequences for the verification of computer security. In particular, we have seen that in imperfect-information game structures and turn-based game structure, effective fairness cannot be expressed in ATL*,

In this chapter, we only considered game structures with two agents. We expect that our results generalize to an unlimited number of agents. The comparison of the expressive power of ATL* and full SSL in turn-based perfect-information game structures is also left for future work. Moreover, we did not consider models with imperfect recall [Sch04b]. It is also not known whether the model checking problems of SSL for models with imperfect recall and concurrent models are decidable.

5

Non-repudiation and Virtual Multi-Protocol Attacks

Abstract. It is well-known that in order for security properties to hold, agents are required to be somehow restricted (for example by not reusing their private keys across protocols). We show that in addition, agents need to have knowledge about the restrictions of other agents. We do so by formally studying the security property of non-repudiation. We investigate of which sub-properties non-repudiation consists, and study the knowledge assumptions that are required for each of them. When these assumption are not satisfied, a new class of attacks, called virtual multi-protocol attacks, arises. In such attacks, the attacker abuses the lack of knowledge of the attacked agent. We demonstrate the practical significance of virtual multi-protocol attacks by providing a case study.

5.1 Introduction

Non-repudiation is, informally, a security property that guarantees that an agent cannot deny having executed some event in a message exchange, if the event has actually occurred in the course of the protocol [ZG96]. One can distinguish *non-repudiation of origin* (NRO), which says that the sender cannot deny having sent a message, and *non-repudiation of receipt* (NRR), which says that the receiver cannot deny having received a message. To achieve non-repudiation, a protocol participant collects evidence, which in case of disagreement does not only convince himself, but also convinces a third party, called the judge [ZG97b]. Evidences are typically implemented using cryptographic signatures.

The fact that protocol participants need to convince an external judge is a major difference between non-repudiation and other security properties, such as secrecy and authentication [Low97]. Moreover, other security properties are typically concerned with external attackers, while non-repudiation is mostly concerned with dishonest behavior of the participants in the protocol itself. In a protocol offering NRR of a message, for instance, the sending agent may try to convince the judge that the message has been received, while in reality, it has not.

Note that, we do not consider *fair exchange* of evidences in this chapter, but only the property of non-repudiation itself. Furthermore, we study the assumptions that are necessary in order to satisfy this definition. In particular, we focus on the knowledge that agents need to have about the behavior of other agents.

In particular, we look at situations where multiple protocols are executed in parallel. Kelsey, Schneier and Wagner [KSW97] have shown that for any protocol

that satisfies a security property if the agents are restricted to executing only that protocol, it is possible to construct a protocol, called the *chosen protocol*, that invalidates this security property if agents execute multiple protocols in parallel. This is caused by the fact that unrestricted agents might send all their private information, including private keys, to the attacker (as the agents might execute another protocol that instructs them to do so). This implies that the attacker can do everything that an unrestricted agent can do. There are multiple ways to prevent this, such as assuming that agents are restricted to executing only one protocol, or using a key for only one protocol. An attack on protocol P that is possible when agents run other protocols at the same time, but not when agents are restricted to executing protocol P, is called a *multi-protocol attack* [KSW97]. Multi-protocol attacks are not only a theoretical problem, but also a realistic threat. Cremers [Cre06] found that out of 30 individually secure protocols from literature, 23 were vulnerable to multi-protocol attacks when executed in parallel with one of the other protocols. A well-known solution to prevent multi-protocol attacks is to assume that agents use different keys for each protocol [GT00]. We will show that this assumption (or at least some restriction on the behavior of agents) is also necessary for non-repudiation.

In addition, in the context of non-repudiation, we identify new necessary assumptions, namely that agents need to be aware about the restrictions of the other agents. This can be seen as an assumption that says that agents *trust* other agents to be restricted (for example, to be restricted to one protocol). If these assumptions are not satisfied, a new class of attacks, called *virtual multi-protocol attacks*, arises.

The remainder of this chapter is organized as follows. In Section 2.3, we give a short introduction to security protocols. In Section 5.2, we give a formal definition of the property of non-repudiation. In Section 5.3, we distinguish between traditional types of non-repudiation, and a newly introduced type of non-repudiation called non-repudiation of intention. In Section 5.4, we study the assumptions that need to be made about how the agents are restricted, and we show that if these assumptions are not satisfied, a new class of attacks, called virtual multi-protocol attacks, arises. In Section 5.5, we demonstrate a virtual multi-protocol attack against an existing protocol, and in Section 5.6, we compare our definition with previous definitions of non-repudiation, before concluding in Section 5.7.

5.2 Non-repudiation

We proceed by formalizing the security property of non-repudiation. A non-repudiation protocol consists of at least three roles: a signer, a relying party, and a judge. The protocol consists of two phases. In the first phase, called the *evidence generation phase*, the signer sends evidence of event e to the relying party. After the first phase, the relying party can optionally start the second phase of the protocol, the *evidence verification phase*, in which he sends his evidence to the judge for verification.

Both phases of the protocol end with a security claim. The evidence generation phase ends with the relying party R claiming *generation of evidence* of event e,

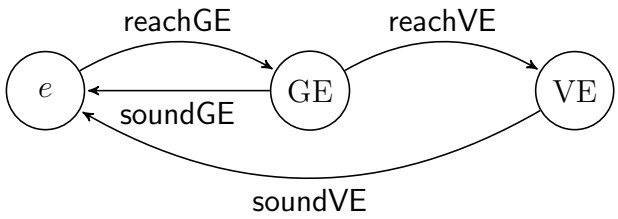


Figure 5.1: Event e and the corresponding two claim events, and the relation between them.

written GE(e), meaning that R is in the possession of evidence that e happened. The evidence verification phase ends with the judge claiming *verification of evidence* of e, written VE(e), meaning that the evidence sufficiently proves to the judge that event e has occurred. The second phase is optional: the relying party can choose not to execute this phase, if no ruling from a judge is necessary.

Below, we will make the meaning of the two security claims precise.

We split the property of non-repudiation into four sub-properties. Each of the two security claims gives rise to two of these properties. The properties are informally depicted in Figure 5.1. The picture shows event e and the corresponding claim events GE and VE, and the relations between these three events. A rightward arrow from one event to another indicates that when the first event has happened, the second event must eventually happen in the future. A leftward arrow means that when the first event has happened, the second event must have happened previously.

The first property is *reachability of GE* (reachGE), which is a functional property with respect to GE(e). This property states that the agent executing role R will eventually reach the claim GE(e). The second property, *soundness of GE* (soundGE), states that R only claims GE(e) if event e has actually happened earlier in the trace. The third property, *reachability of VE* (reachVE), states that if R claims GE(e), then at some point later in the trace, J will claim VE(e). The fourth property, *soundness of VE* (soundVE), states that if the judge claims VE(e), then e has actually happened.

Note that soundGE is implied by reachVE and soundVE. However, we still consider soundGE individually, because the assumptions for each of the properties differ, as we will see in Section 5.4.

If we assume that different agents might behave in different ways, whether the property holds or not is dependent on which agents run which roles. Therefore, we parametrize the security properties with a role assignment.

Definition 5.1 (Reachability of GE, Soundness of GE, Reachability of VE, Soundness of VE)**.** *Given a trace t and a role assignment ρ, we define the following four security properties:*

- *Reachability of GE in trace t under role assignment ρ, written $\mathsf{reachGE}_t(\rho)$, holds whenever $t_i = ((\theta, \rho, \sigma), e)$ for some θ and σ, $active(t_i) = S$, there exists $j > i$ and an instantiation $inst'$ such that $t_j = (inst', \mathsf{claim}(R, GE(e)))$.*
- *Soundness of GE in trace t under role assignment ρ, written $\mathsf{soundGE}_t(\rho)$,*

holds if whenever $t_i = ((\theta, \rho, \sigma), \mathsf{claim}(R, GE(e)))$ *for some* θ *and* σ*, there exists* $j < i$ *and some instantiation* $inst'$ *such that* $t_j = (inst', e)$*.*

- *Reachability of VE in trace* t *under role assignment* ρ*, written* $\mathsf{reachVE}_t(\rho)$*, holds whenever* $t_i = ((\theta, \rho, \sigma), \mathsf{claim}(R, GE(e)))$ *for some* θ *and* σ*, there exists* $j > i$ *and some instantiation* $inst'$ *such that* $t_j = (inst', \mathsf{claim}(J, VE(e)))$*.*
- *Soundness of VE in trace* t *under role assignment* ρ*, written* $\mathsf{soundVE}_t(\rho)$*, holds whenever* $t_i = ((\theta, \rho, \sigma), \mathsf{claim}(J, VE(e)))$ *for some* θ *and* σ*, there exists* $j < i$ *and some instantiation* $inst'$ *such that* $t_j = (inst', e)$*.*

We illustrate this definition with an example.

Example 5.1. *We assume that all agents are restricted to executing only this protocol, and that agents do not indefinitely postpone executing available actions. Protocol 5.1 satisfies all security properties of Definition 5.1, given event* $e = \mathsf{send}(S, R, \{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)})$*. The evidence generation phase consists of steps (i) and (ii), while the evidence verification phase consists of steps (iii) and (iv). The constants* f_1 *and* f_2 *are called* flags*, and are used to prevent type-flaw attacks [HLS03].*

Protocol 5.1

1. $S \to R$: $\{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)}$
2. $\mathsf{claim}(R, \mathrm{GE}(e))$
3. $R \to J$: $\{\!|\, f_2, \{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)} \,|\!\}_{\mathsf{pk}(R)}$
4. $\mathsf{claim}(J, \mathrm{VE}(e))$

We proceed by (informally) checking that this protocol satisfies all four properties. When step (i) and (ii) are executed, $\mathsf{claim}(R, GE(e))$ *is reached, so* $\mathsf{reachGE}$ *is satisfied. For* $\mathsf{soundGE}$*, we observe the following. Whenever* R *executes* $\mathsf{claim}(R, GE(e))$*, he must have received* $\{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)}$*. As* $\{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)}$ *can only be generated by* S*, the event* $\mathsf{send}(S, R, \{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)})$ *must have been executed. Next, we check* $\mathsf{reachVE}$*. If* R *claims GE of* e*, steps (iii) and (iv) lead to* J *reaching* $\mathsf{claim}(J, VE(e))$*. Finally, if* J *claims VE of* e*, then step (iv) is executed, and he only executes step (iv) after he receives* $\{\!|\, \{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)} \,|\!\}_{\mathsf{pk}(R)}$*, due to the use of the signature. Only* S *could have generated* $\{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)}$*, so* $\{\!|\, f_1, R, J, m \,|\!\}_{\mathsf{pk}(S)}$ *must have been sent by* S*, which implies* $\mathsf{soundVE}$*. This implies that Protocol 5.1 is a correct non-repudiation protocol under the above-mentioned assumptions.*

5.3 Non-repudiation of Intention

Non-repudiation of origin (NRO) states that an agent cannot deny having sent a message. In many applications, however, we need a stronger property, namely that an agent cannot deny having had a certain *intention* with sending a message. We take contract signing as an example.

In contract signing, NRO is necessary, as it should be required that it is impossible for an agent to deny having put a signature on a contract. However, we will see that we need an even stronger property than NRO. It is important to realize that the word *signing* is used for two entirely different concepts. *Cryptographic signing*, obtained by public key cryptography, guarantees that a message originates at a certain agent. *Legal signing*, such as a hand-written signature on paper, indicates that the agent agrees with the content of the contract. These two definitions do not necessarily correspond.

One of the requirements of a contract signing scheme (in the legal sense) should be that an agent that follows the protocol never signs a contract without intending to do so. That means that in contract signing, it should not only be required that a signature on a contract is originated by a certain agent, but also that this agent has done so with the intention of signing a contract. NRO does not accomplish this, which implies that NRO is not sufficient for contract signing, as sometimes is assumed [GM99, ODSS04, AE11].

As NRO is not suitable for applications that require intention, we introduce a new security property, called *non-repudiation of intention* (NRI). We model this by adding so called intention events, that have the form $\mathsf{intend}(S, Int, rt)$, and denote that S has the intention to commit to the content of rt. We only model this event informally. We do this by requiring that S only executes $\mathsf{intend}(S, Int, rt)$ if he has the intention to commit to rt. Then we define NRI as non-repudiation of the intention event.

Definition 5.2 (Intention event, non-repudiation of intention)**.** *If S is a role, Int is an intention, and rt is a role term, then* $\mathsf{intend}(S, Int, rt)$ *is an* intention event*, denoting that S intends rt.*

A protocol guarantees non-repudiation of intention *(NRI) if there exists an intention event e such that non-repudiation of e is guaranteed.*

The following example illustrates the difference between NRO and NRI.

Example 5.2. *Consider the following non-repudiation protocol with event $e =$ $\mathsf{send}(S, R, m)$ and evidence $NR(e) = \{\!|\, m \,|\!\}_{\mathsf{pk}(S)}$. In this protocol, n is a nonce.*

Protocol 5.2

1. $R \to S$: n
2. $S \to R$: $\{\!|\, n, R, J \,|\!\}_{\mathsf{pk}(S)}$
3. $R \to S$: m
4. $S \to R$: $\{\!|\, m, R, J \,|\!\}_{\mathsf{pk}(S)}$

One can easily check that $\{\!|\, m, R, J \,|\!\}_{\mathsf{pk}(S)}$ is indeed a proof that S has sent the contract m. However, the following trace proves that the evidence $\{\!|\, m, R, J \,|\!\}_{\mathsf{pk}(S)}$ cannot be used as an evidence that S signed the contract m (in the legal sense).

Trace: $a[\mathit{Protocol\ 5.2} : \rho(R) = b, \rho(S) = a)]; b[\mathit{unrestricted}]$.

1. $b \to a$: m
2. $a \to b$: $\{\!|\, m, R, J \,|\!\}_{\mathsf{pk}(a)}$

In this trace, b sends a contract m instead of a nonce n as the first message of the protocol. Note that the protocol does not specify that m has a special structure, so a cannot differentiate between a nonce and a contract. Therefore a expects that she has been sent a nonce and signs it. Consequently, b possesses evidence $\{\!|\, m, R, J \,|\!\}_{\mathsf{pk}(b)}$*, while a did not intend to sign a contract. This shows that this protocol cannot be used for contract signing (although it can be used for NRO).*

Protocols that offer non-repudiation can not in general be used for any application where it is required that an agent cannot deny his intention. For example, in certified e-mail [ZG97a], NRR guarantees that a message has been received. However, there is no guarantee that the receiving agent had the intention of receiving an e-mail message. For example, while receiving the certified e-mail, the agent might have thought he only received a nonce, and might have discarded it immediately.

To formally show that Protocol 5.2 cannot be used for NRI, we add an event expressing that S has the intention to agree to a contract at the beginning of the protocol, and change R's claim in NRI of this event. This results in the following protocol.

Protocol 5.3

1. $e = \mathsf{intend}(S, \text{agree to contract } m, m)$
2. $R \to S$: n
3. $S \to R$: $\{\!|\, n, R, J \,|\!\}_{\mathsf{pk}(S)}$
4. $R \to S$: m
5. $S \to R$: $\{\!|\, m, R, J \,|\!\}_{\mathsf{pk}(S)}$
6. $\mathsf{claim}(R, \mathrm{GE}(e))$

This protocol does not satisfy soundness of VE, as S might use $\{\!|\, n, R, J \,|\!\}_{\mathsf{pk}(S)}$ as evidence, while S never agreed to n, proving that this protocol cannot be used for NRI.

To further illustrate the difference between NRI and other forms of non-repudiation, we consider a vulnerability which we recently discovered in a software package called Request Tracker.

Example 5.3. *Request Tracker [Pra] is a web and email-based trouble ticketing system, which is developed by Best Practical. Trouble ticketing systems are used by an organization's customer support call center to manage reported customer issues. Request Tracker is included in the Ubuntu universe repository, and widely used (for example by OpenSSL, Verisign, and NASA). It is possible to set up Request Tracker to sign all outgoing e-mail. Furthermore, the default behavior of Request Tracker is such that if the system receives an incoming e-mail for a certain trouble ticket,*

it automatically forwards the unaltered message to all e-mail addresses that are listed as cc for this ticket. This can be abused in the following way. The attacker asks the system owner for a ticket to be created, with a second e-mail address controlled by the attacker as a cc. Subsequently, the attacker sends an e-mail message concerning this ticket to Request Tracker, and receives a version of his e-mail in return that is signed with the PGP key of the system owner. This implies that an Request Tracker system with PGP-signing of outgoing e-mail enabled can be used as a signing oracle. Users of Request Tracker might assume that e-mail signed by Request Tracker implies NRI, and has a meaning comparable with physical signature. The designers of Request Tracker, though, apparently only had in mind that the signatures would offer NRO. The author has demonstrated this vulnerability on a Luxembourgish government agency. Best Practical has been contacted by the authors, and subsequently issued a patch to their system [CVE]. The government agency has been contacted as well.

In practice, implementations of non-repudiation, such as PGP, do not make the distinction between NRO and NRI explicit. The vulnerability in Request Tracker shows that this might lead to security problems.

5.4 Assumptions of Non-repudiation

5.4.1 Assumptions about Restrictions on Agents

For the usual security properties, such as security and authentication, it is normally assumed that the communication partners are restricted to executing only that protocol [CM03, CM12]. Sometimes such restricted agents are called 'honest' or 'trusted'. For security and authentication, this requirement is understandable, because for such security properties, the participants in the run have an incentive to validate the security claim.

However, for non-repudiation, the assumption that all communication partners are restricted is too strong. This is because in the case of non-repudiation, the communication partners have an incentive to *invalidate* the security claims of the other agents. For example, a non-repudiation protocol should protect against the relying party trying to convince the judge with false evidence. This implies that non-repudiation protocols also need to protect against attacks from the communication partners that deviate from the protocol description.

However, in order to be able to make any claims, at least some restrictions on agents must be assumed. If agents are not restricted at all, no security properties can be satisfied in any protocol [KSW97]. We define which restrictions on agents we assume for each of the sub-properties of non-repudiation. The first property we look at is soundGE. Remember that this property states that R only claims $\mathrm{GE}(e)$ if e has actually happened earlier in the trace. If the signer is unrestricted, he might give away all his private information to Eve, an agent controlled by the attacker. It is therefore impossible to distinguish between an event executed by the signer and an event executed by Eve.

This shows that in most practical contexts, the most a protocol can offer is the guarantee that an agent either executed an event or was not restricted to the pro-

tocol. Any requirement beyond that should be solved by non-technical measures such as a legal system in which agents are liable for any consequence of not following the protocol. For example, in contract signing, this means that the gain of ignoring a contract should never be larger than the punishment for not following the protocol. The details of such non-technical measures are outside the scope of this chapter. Therefore, in the rest of this chapter, we may assume that the signer is restricted.

Furthermore, if R is unrestricted, he might make a claim at a moment when it is not (yet) appropriate. Therefore, in the context of soundGE, R should be assumed to be restricted as well.

Note that since S is assumed to be restricted, the relation between the evidence and the event can be arbitrary. For example, Protocol 5.4 with $e = \mathsf{send}(R, S, m)$ and evidence $\mathrm{NR}(e) = \{\!|\, \mathsf{h}(m) \,|\!\}_{\mathsf{pk}(S)}$ is a correct non-repudiation protocol:

The assumptions for the second property, soundVE, are similar: we need to assume that R and S are unrestricted.

We proceed by looking at the assumptions for reachGE. This property can in general not be satisfied if S is unwilling to generate the evidence, or when R is unwilling to receive the evidence. Therefore, reachGE cannot be satisfied when either S or R are unrestricted. Similarly, to achieve reachVE, R should be willing to send evidence, and J should be willing to receive it. This implies that reachVE cannot be satisfied when either R or J is unrestricted.

In order to reason about such restrictions on agents, we introduce a logic. This logic contains all usual propositional connectives, atomic formulas of the form $\mathrm{Res}_t(a, P)$, indicating that in trace t, agent a is restricted to protocol P, and atomic formulas of the form $\mathrm{Resilient}_t(a, P)$, indicating that in trace t, agent a is resilient when executing protocol P.

The properties reachGE and reachVE require the existence of a future event. Therefore, we need to assume resilient agents (see Section 2.3.5). For reachGE, we need to assume that S and R are resilient, and for reachVE, we need to assume that R and J are resilient.

One possible restriction on agents is the restriction of agents to only one protocol. Formally, we say that an agent is *restricted* to protocol P, if the agent only creates roles of protocol P.

Definition 5.3 (Restricted to protocol)**.** *Agent a is restricted to protocol P in trace t, written $Res_t(a, P)$, if for all t_i such that $actor(t_i) = a$, we have $protocol_t(t_i) = P$.*

We have seen that for each of the properties, some agents must be assumed to be restricted. If we assume that these restricted agents are restricted to executing only the protocol in question, we obtain the following definition.

Definition 5.4 (Satisfying non-repudiation)**.** *A protocol P satisfies non-repudiation if for all role assignments ρ and all traces t, we have:*

1. $Res_t(\rho(S), P) \wedge Res_t(\rho(R), P) \wedge Resilient_t(\rho(S), P) \wedge Resilient_t(\rho(R), P) \rightarrow \mathsf{reachGE}_t(\rho)$.
2. $Res_t(\rho(S), P) \wedge Res_t(\rho(R), P) \rightarrow \mathsf{soundGE}_t(\rho)$.

3. $Res_t(\rho(R), P) \wedge Res_t(\rho(J), P) \wedge Resilient_t(\rho(R), P) \wedge Resilient_t(\rho(J), P) \rightarrow \mathsf{reachVE}_t(\rho)$.
4. $Res_t(\rho(S), P) \wedge Res_t(\rho(J), P) \rightarrow \mathsf{soundVE}_t(\rho)$.

Example 5.1 satisfies non-repudiation under protocol-restriction assumptions, which shows that non-repudiation can indeed be satisfied under this definition.

Now let us have a look at which attack scenarios this definiton prevents. First of all, all attacks on reachGE, soundGE, reachVE and soundVE from outside agents, i.e., agents not participating in the protocol run, are prevented. Moreover, requirement (1) prevents attacks from the judge against reachGE, i.e., the judge cannot prevent agents from reaching GE(e); Requirement (2) prevents attacks from the judge against soundGE, i.e., the judge cannot trick the relying agent into believing false evidence; Requirement (3) prevents attacks from the signer against reachVE, i.e., the signer cannot prevents the relying agent from convincing the judge; Requirement (4) prevents attacks from the relying agent against soundVE, i.e., the relying agent cannot cause the judge to make a false claim. The final requirement is what is typically seen as the most important requirement of non-repudiation.

Note that two attack scenarios cannot be prevented. First, we do not prevent attacks from the signer against soundGE, i.e., the signer might cause the relying agent to make a false claim; Second, we do not prevent attacks from the signer against soundVE, i.e., the signer might cause the judge to make a false claim.

Protocol 5.4

1. $R \rightarrow S$: $\mathsf{h}(m)$
2. $R \rightarrow S$: m
3. $S \rightarrow R$: $\{\!|\, R, J, \mathsf{h}(m) \,|\!\}_{\mathsf{pk}(S)}$
4. $\mathsf{claim}(R, \mathrm{GE}(e))$
5. $R \rightarrow J$: $\{\!|\, \{\!|\, R, J, \mathsf{h}(m) \,|\!\}_{\mathsf{pk}(S)} \,|\!\}_{\mathsf{pk}(R)}$
6. $\mathsf{claim}(J, \mathrm{VE}(e))$

Alternatively, if a protocol can only be used for agreeing to a constant (for example, the text 'I sell you my house'), a signed empty string can be used as evidence. Even a protocol where evidence 'I do not agree to m' is evidence of agreeing to m might be correct.

Assuming that agents are restricted to one protocol is a strong restriction, as in real-life situations, agents might run several protocols (perhaps with different purposes) at the same time. We can also assume weaker restrictions that allow that agents execute multiple protocols in parallel, for example by assuming that agents use a different key for every protocol (see e.g. [GT00]). However, the choice of the exact restriction is not the main point. The main point is that there need to be at least *some* way in which the agents are assumed to be restricted.

Another way to avoid multi-protocol attacks is to make use of *disjoint encryption* [GT00], meaning that when one protocol uses a particular form of encrypted mes-

sages or signatures, the other protocols must not construct a message of the same form. One way to arrange for this is to assign each protocol a unique identifier, and include this identifier in each encryption. This is not an ideal solution, as it involves altering the protocols themselves. Moreover, if the set of possible protocols is not known in advance, it cannot be guaranteed that there do not exist two protocols that tag messages in the same way. Therefore, an easier way to achieve disjoint encryption is to ensure that agents use a different key for every protocol [GT00].

Definition 5.5 (Restricted to protocol with key)**.** *Agent a is restricted to protocol P in trace t when using key k, written $Res_t(a, P, k)$, if for all t_i such that $actor(t_i) = a$ and $protocol_t(t_i) = P$, then k is not in the initial knowledge of that run.*

Now we can define what it means for a protocol to satisfy non-repudiation under

Definition 5.6 (Satisfying non-repudiation under protocol-key assumptions)**.** *A protocol satisfies non-repudiation under protocol-key assumptions if for all role assignments ρ and pairwise disjoint keys k_S, k_R, k_J we have:*

1. $Res_t(\rho(S), P, k_S) \wedge Res_t(\rho(R), P, k_R) \wedge Resilient_t(\rho(S), P) \wedge Resilient_t(\rho(R), P)$ $\rightarrow \mathsf{reachGE}_t(\rho)$.
2. $Res_t(\rho(S), P, k_S) \wedge Res_t(\rho(R), P, k_R)$ $\rightarrow \mathsf{soundGE}_t(\rho)$.
3. $Res_t(\rho(R), P, k_R) \wedge Res_t(\rho(J), P, k_J) \wedge Resilient_t(\rho(R), P) \wedge Resilient_t(\rho(J), P)$ $\rightarrow \mathsf{reachVE}_t(\rho)$.
4. $Res_t(\rho(S), P, k_S) \wedge Res_t(\rho(J), P, k_J)$ $\rightarrow \mathsf{soundVE}_t(\rho)$.

Protocol 5.5 can be used for non-repudiation when executed in isolation. However, protocols are typically executed in a context in which many protocols are ran in parallel. In Example 5.4, we will show a multi-protocol attack on Protocol 5.5, i.e., we will show that this protocol is correct when executed in isolation, but not when executed in parallel with some other protocol.

Example 5.4. *Protocol 5.5 is a non-repudiation protocol in which event $e = \mathsf{send}(R, S, m)$ and evidence $NR(e) = \{\!|\, m \,|\!\}_{\mathsf{pk}(S)}$. Further, consider authentication Protocol 5.6, which functions as a chosen protocol that attacks Protocol 5.5.*

Protocol 5.5

1. $R \rightarrow S$: m
2. $S \rightarrow R$: $\{\!|\, m \,|\!\}_{\mathsf{pk}(S)}$

Protocol 5.6

1. $P \rightarrow Q$: n
2. $Q \rightarrow P$: $\{\!|\, n \,|\!\}_{\mathsf{pk}(Q)}$

When Protocols 5.5 and 5.6 are executed in parallel, b can execute Protocol 5.6, with a in role Q and b in role P. However, b can be unrestricted and send a contract m instead of a nonce. Because a expects a nonce, she will sign it, not realizing that

by doing that, she gives b evidence that she signed the contract m. It is clear that this situation is undesirable in the context of contract signing, as it does not satisfy the property that if a contract is signed, it should be signed intentionally.

So far we only studied attacks in which the attack is executed by the receiving party R. However, there also exist attacks on a protocol P in which both the signer and relying party are restricted to P, but where a third party can execute the attack.

Example 5.5. *If Protocols 5.5 and 5.6 are executed in parallel, the following attack trace becomes possible.*

Trace: $a[\mathit{Protocol\ 5.6} : \rho(S) = a, \rho(R) = e)]; b[\mathit{Protocol\ 5.5} : \rho(S) = a, \rho(R) = b)]; e[\mathit{unrestricted}]$.

1. $b \rightarrow e$: m
2. $e \rightarrow a$: m
3. $a \rightarrow e$: $\{\!|\, m \,|\!\}_{\mathsf{pk}(R)}$
4. $e \rightarrow b$: $\{\!|\, m \,|\!\}_{\mathsf{pk}(R)}$

This means that b obtains evidence that a originated contract m. However, a did not intend to sign contract M. Note that this problem cannot be solved by adding a layer of authentication (so that b can verify that he is indeed speaking with a). That is because E can alter the chosen protocol in such a way that he can obtain the required signatures from a.

5.4.2 Virtual Multi-Protocol Attacks

We have implicitly assumed that the restrictions on agents are *common knowledge*. With that, we mean that the restrictions are known by all agents, and all agents know that they are known, and all agents know that all agents know that they are known, etc. For example, we assumed that agents have full knowledge about how agents are restricted to protocols. This assumption is, however, not necessarily realistic. For example, in networks where agents do not know their communication partners in advance, such as in ad-hoc networks or the internet, agents have no prior knowledge about the restrictions on the other agents. Note that we are not talking about dynamic knowledge about the current state of the agent (i.e. [ZG98, KR10]), but about static knowledge concerning the restrictions on the behavior of other agents, that does not change during the course of the protocol execution.

We make our model more realistic by dropping the assumption that agents have common knowledge about restrictions on behavior of agents. We do so by explicitly modeling the knowledge that agents have about the restrictions on the behavior of other agents.

The reason that knowledge on the restrictions on agents is important, is that a security property should not only hold, but also be known to hold. When restrictions on agents are not common knowledge, the fact that a security property holds, does

not automatically imply that an agent *knows* that the property holds, as different agents might consider different sets of possible traces.

We proceed by studying for each subproperty of non-repudiation which agents should know that this property holds. First of all, R must know that soundGE holds. If R does not know that soundGE holds, R cannot be sure that when he reaches the GE claim, the event indeed happened. This is problematic: R must be convinced about the validity of the evidence. Note that the fact that soundGE might actually hold, changes nothing about this fact. The relevant factor is whether R knows that soundGE holds.

For the same reason, J should know that soundVE holds.

Furthermore, R should know that reachVE holds. If R does not know that reachVE holds, he cannot rely on the evidence, as he is not sure that he will be able to convince J.

However, the knowledge of the properties is not sufficient: agents might even need *knowledge of knowledge* of the properties. Consider that if R has knowledge of the fact that reachVE holds, he knows that J will reach the VE claim. However, R does not know whether J's VE claim implies that J is convinced that e has happened. Therefore, in order for R to have his evidence accepted, R needs to know that J knows that soundVE holds.

The fact that agents need to know that a security property holds has as a consequence that the correctness of a non-repudiation protocol does not only depend on the restrictions on agents (as is well-known), but also on the knowledge that agents have about these restrictions. To see that lack of knowledge about the restrictions of other agents might invalidate security properties that held previously, consider the following example.

Example 5.6. *If R knows that S is restricted to Protocol 5.1, we can easily check that S knows that soundGE holds. However, if R does not know of this restriction, R does not know that soundGE holds. This is because R must consider the fact that S might execute a protocol in which S sends out his signing key k_S to a third party, which has as a consequence that the message $\{\!|\, f_2, m \,|\!\}_{\mathsf{pk}(k_S)}$ is not guaranteed to originate at S.*

Agents might even need *knowledge of knowledge* about the restrictions. The following example informally illustrates this.

Example 5.7. *Assume that R, J and S are restricted to Protocol 5.1, and that all agents know that all other agents are restricted to this protocol. Assume that R does not know that J knows that S is restricted to Protocol 5.1. If J does not know that S is restricted to Protocol 5.1, then J does not know that soundVE holds (as J might consider that S executes a protocol that instructs S to send out his signing key). Therefore, if R does not know that J knows that S is restricted to Protocol 5.1, R does not know whether J knows soundVE.*

To reason about the knowledge that agents have about each other's restrictions, and the knowledge about this knowledge etc., we develop a possible world semantics [FHMV95], and a corresponding logic. This logic is interpreted in *protocol-restriction models*, which consist of a set of worlds, a property *Restricted*, and a

property *Progress* that determines the set of worlds in which an agent is restricted to a protocol, and resilient in a protocol, respectively, and for each agent a relation on the set of worlds that indicates which worlds are indistinguishable to that agent.

Definition 5.7 (Protocol-restriction model)**.** *A protocol-restriction model is a tuple* $(W, Restricted, Progress, \sim_1, \ldots, \sim_n)$*, where:*

- W *is a set of worlds, where each world defines a set of traces;*
- *given agent* a *and protocol* P*, we say that* $Restricted(a, P) \subseteq W$ *is the set of worlds where* a *only executes protocol* P*;*
- *given agent* a *and protocol* P*, we say that* $Progress(a, P) \subseteq W$ *is the set of worlds where* a *is resilient;*
- $\sim_1, \ldots, \sim_n$ *are equivalence relations on* W *for* n *number of agents. If* $w \sim_i w'$*, we say that* w *and* w' *are indistinguishable for agent* i*.*

Whenever $w \sim_a w'$ *for agent* a*, then* $w \in Restricted(a, P)$ *implies that* $w' \in Restricted(a, P)$*, and* $w \in Progress(a, P)$ *implies that* $w' \in Progress(a, P)$*.*

The final property specifies that agents know their own restrictions.

A world w in model M defines a set of possible traces, which we indicate by $Tr(M, w)$.

Definition 5.8 (Traces of world in protocol-restriction model)**.** *The set of traces of world* w *in model* $M = (W, Restricted, Progress, \sim_1, \ldots, \sim_n)$*, written* $Tr(M, w)$*, is the set of traces* t *such that for every agent* a *and every protocol* P*, we have that* $w \in Restricted(a, P)$ *implies* $Res_t(a, P)$*, and* $w \in Progress(a, P)$ *implies that we have* $Resilient_t(a, P)$*.*

The semantics of the logic is defined as follows, given a model M and a world w.

Definition 5.9 (Interpretation of protocol-restriction models)**.** *Let* a *be an agent and let* P *be a protocol. We define the interpretation of a formula in a protocol-restriction model* M *and a world* w *as follows.*

- $M, w \models K_a\varphi$ *if for all* w' *such that* $w \sim_a w'$*, we have* $M, w' \models \varphi$*.*
- $M, w \models \pi(\rho)$ *for property* π *in* $\{\mathsf{reachGE}, \mathsf{soundGE}, \mathsf{reachVE}, \mathsf{soundVE}\}$ *and role assignment* ρ*, if for all traces* $t \in Tr(M, w)$*, we have we have* $\pi_t(\rho)$*.*
- $M, w \models Res(a, P)$ *if* $w \in Restricted(a, P)$*.*
- $M, w \models Resilient(a, P)$ *if* $w \in Progress(a, P)$*.*

This definition says that an agent knows φ in world w, written $M, w \models K_a\varphi$, if it is true in every world that he cannot distinguish from that world. This implies that an agent only knows things that are true. Furthermore, a property holds in a world if it holds in all traces in that world. The last two properties are straightforward.

Given a formula φ, we write $\models \varphi$ whenever for all models M and all worlds w of M, it holds that $M, w \models \varphi$.

Now we will study which knowledge of agents need to be assumed in order to satisfy the knowledge of the security property. The following property of this logic states that agents cannot know a security property without knowing the assumption that is required to make this security property true.

Property 5.1. *If φ is K-free, and $\pi \in \{\mathsf{reachGE}, \mathsf{soundGE}, \mathsf{reachVE}, \mathsf{soundGE}\}$, and ρ is a role assignment, then we have that $\not\models \pi(\rho)$ implies*

$$\not\models K_{a_1} \dots K_{a_{k-1}} K_{a_{k+1}} \dots K_{a_n} \varphi \rightarrow K_{a_1} \dots K_{a_n} \pi(\rho).$$

Now we can define non-repudiation in protocol-restriction models as follows.

Definition 5.10 (Satisfying non-repudiation in protocol-restriction models)**.** *A protocol P satisfies non-repudiation in protocol-restriction models if for all role assignments ρ, we have:*

1. $\models Res(\rho(S), P) \wedge Res(\rho(R), P) \wedge Resilient(\rho(S), P) \wedge Resilient(\rho(R), P)$
   $\rightarrow \mathsf{reachGE}(\rho)$.
2. $\models Res(\rho(S), P) \wedge Res(\rho(R), P)$
   $\rightarrow \mathsf{soundGE}(\rho)$.
3. $\models Res(\rho(R), P) \wedge Res(\rho(J), P) \wedge Resilient(\rho(R), P) \wedge Resilient(\rho(J), P)$
   $\rightarrow \mathsf{reachVE}(\rho)$.
4. $\models Res(\rho(S), P) \wedge Res(\rho(J), P)$
   $\rightarrow \mathsf{soundVE}(\rho)$.
5. $\models K_{\rho(R)}(Res(\rho(S), P) \wedge Res(\rho(R), P))$
   $\rightarrow K_{\rho(R)}\mathsf{soundGE}(\rho)$.
6. $\models K_{\rho(R)}(Res(\rho(R), P) \wedge Res(\rho(J), P) \wedge Resilient(\rho(R), P) \wedge Resilient(\rho(J), P))$
   $\rightarrow K_{\rho(R)}\mathsf{reachVE}(\rho)$.
7. $\models K_{\rho(J)}(Res(\rho(S), P) \wedge Res(\rho(J), P))$
   $\rightarrow K_{\rho(J)}\mathsf{soundVE}(\rho)$.
8. $\models K_{\rho(R)}K_{\rho(J)}(Res(\rho(S), P) \wedge Res(\rho(J), P))$
   $\rightarrow K_{\rho(R)}K_{\rho(J)}\mathsf{soundVE}(\rho)$.

Property 5.2. *For every agent a, propositional formula α and formula Ψ, it holds that $\models \alpha \rightarrow \Psi$ implies $\models K_a\alpha \rightarrow K_a\Psi$.*

Proof. Let a be an agent, let α be a propositional formula, and let Ψ be a formula. Assume $\models \alpha \rightarrow \Psi$. Let M be a protocol-restriction model and let w be a world. Assume $M, w \models K_a\alpha$. Let w' be a world such that $w' \sim_a w$. Since $M, w \models K_a\alpha$, we have $M, w' \models \alpha$. Since $\models \alpha \rightarrow \Psi$, we have $M, w' \models \alpha \rightarrow \Psi$, and thus $M, w' \models \Psi$. Therefore, $M, w \models K_a\Psi$. This shows that $M, w \models K_a\alpha \rightarrow K_a\Psi$, so we have $\models K_a\alpha \rightarrow K_a\Psi$. □

Now we show that Protocol 5.1 satisfies item 1 of Definition 5.10. In Section 5.4.1, we saw that it satisfies $(\mathrm{Res}(\rho(S), P, k_S) \wedge \mathrm{Res}(\rho(R), P, k_R)) \rightarrow \mathsf{soundGE}(\rho)$. Hence it also satisfies $\models K_{\rho(R)}(\mathrm{Res}(\rho(S), P, k_S) \wedge \mathrm{Res}(\rho(R), P, k_R)) \rightarrow \mathsf{soundGE}(\rho)$, and thus item 5 is also satisfied. It can be shown similarly that the protocol satisfies the other properties.

Property 5.2 shows that there exist protocols that satisfy non-repudiation in protocol-restriction models. For example (5): In Section 5.2, we saw that there exist protocols that satisfy $\models (\mathrm{Res}(\rho(S), P, k_S) \wedge \mathrm{Res}(\rho(R), P, k_R)) \rightarrow \mathsf{soundGE}(\rho)$. Therefore, by Property 5.2, there exist protocols that satisfy the other assumptions as well.

From definition 5.10, it follows that the following assumptions need to be made in order to allow protocols that satisfy non-repudiation.

1. S, R and J are restricted to the protocol and resilient.
2. R knows that S, R and J are restricted to the protocol.
3. J knows that S and J are restricted to the protocol.
4. R knows that J knows that S and J are restricted to the protocol.
5. R knows that S and J are resilient.

Again, we can replace the restriction to the protocol by other restrictions. For example, one could instead assume that the agent has a key that he uses only for one protocol.

In that case, the first assumption corresponds to the standard requirement that agents should not re-use their signing key across protocols. However, to the best of the authors' knowledge, the other assumptions have not been formulated before. When these latter restrictions are not satisfied, a new type of attack arises, which we call *virtual multi-protocol attacks*. We speak of a virtual multi-protocol attack when a security property holds in a world of a model, but when some agent does not *know* that the property holds.

Definition 5.11 (Virtual multi-protocol attack)**.** *A world w in protocol-restriction model M is vulnerable to a* virtual multi-protocol attack *on security property φ against agent a if $M, w \models \varphi$ and $M, w \not\models K_a\varphi$.*

We call the attack *virtual* because the attacker does not need to execute a role that invalidates the security property, but only needs to imply that he might have done so. Now we discuss some possible ideas to prevent virtual multi-protocol attacks that do not work. For example, if the judge does not know about the restrictions on the signer, the signer might correctly execute a non-repudiation protocol, and afterwards claim that he in fact executed a chosen protocol that is especially crafted to invalidate the security property of the first protocol (for example a protocol that makes the signer's signing key available to others). The signer might either do this deliberately, or accidentally when he is tricked by the attacker into executing such a protocol.

Note that virtual multi-protocol attacks cannot be prevented by standard techniques to avoid multi-protocol attacks or type flaw attacks. In particular, it is not sufficient to assume that all protocols are tagged with a protocol identifier, or that each message is tagged with a flag indicating the type of the message, as proposed in [HLS03]. This is because R and J do not know that the signer only executes correctly tagged or flagged protocols. As the attacker can choose the chosen protocol, the attacker can select a chosen protocol that tags or flags messages in the same way as the original protocol. Finally, a virtual multi-protocol attack cannot be prevented by requiring that agents use a different key for different protocols, as long as the relying party and judge do not *know* that agents use a different key for different protocols. A protocol especially crafted to serve as chosen protocol against another protocol might be trivial, and claiming to be the victim of a multi-protocol attack involving a trivial protocol might not be credible. However, less contrived chosen protocols can be created as well, as we will see in Section 5.5.

We have seen that agents need to be assumed to have knowledge about each other's restrictions, for example in the form of knowledge about which key is used for which protocol. Note that this knowledge cannot be achieved through a cryptographic protocol. If this were possible, we could prepend the non-repudiation protocol with a protocol that achieves the required knowledge, and obtain a non-repudiation security protocol in which no prior knowledge is required. For example, it is not sufficient when an agent sends a message that says 'I execute only non-repudiation protocol P' to the other parties, as the agent might later deny having originated such a message. In other words, we also need NRO of this message, which leads to a bootstrapping problem.

The knowledge assumption can only be satisfied when agents communicate their restrictions to each other. This process should happen by means of a communication medium that does not suffer from the attacks mentioned above, i.e., in a non-digital way. As it is not practical for the judge to have a personal interaction with all agents that execute non-repudiation protocols, this could be implemented by using key servers that spread information about restrictions on agents. For example, the restrictions about which keys are used for which protocols could be propagated by making the key server not only link public keys and user identities (as normally happens), but additionally link the key to the exact protocol (and version) for which it will be used. Alternatively, certificates can be used for the same purpose. Again, it is required that the interaction between the signer and the key server or certificate authority happens along non-digital way, in order to prevent the agent from denying that he has deposited his key for a certain protocol.

Attacks on NRI protocols can occur in a multi-protocol context as well. As we have seen, for contract-signing (or any protocol requiring non-repudiation of intention), agents are required to have knowledge about each other's restrictions as well. In particular, if the relying party does not know for which protocols the signer of the contract uses his key, virtual multi-protocol attacks are possible: The relying party can always deny having intended to sign the contract by claiming to have executed a protocol in which the evidence of the contract signing protocol has a different interpretation.

The protocol that the signer claims to have executed in a virtual multi-protocol attack is not required to be an existing or realistic protocol. However, the relying

party (and the judge) should believe that the signer might have executed such a protocol. Therefore, it helps if the executed protocol looks credible. In the next section, we will look at three examples of credible looking protocols that can be used as virtual multi-protocol attack against published protocols.

Furthermore, note that the attack cannot be prevented by requiring that contracts are always messages with a certain structure, and that messages with this structure should not be signed without agreeing to the contract. In other words, it should be prohibited to sign a message that has the structure of a contract without agreeing to it. There is currently no such standard, and even if such a standard were available, an agent could always deny to be aware of the standard.

Consider protocol P where message m is evidence of the origin of m, and protocol P' where message $h(m)$ is evidence of the origin of m. We assume that messages and hashes are not distinguishable (one can think, for example, of the message as a random number). If the relying party does not know that the signer uses his signing key only for protocol P, the signer can execute protocol P, giving the relying party evidence of the origin of m. However, the signer can deny the validity of the evidence by claiming that the evidence originated from protocol P', even without actually having executed P'. This shows why we need $K_{\rho(R)}$ as condition of soundGE, since under this assumption, R can exclude that S used the key with which his evidence is signed in P'. Similarly, we need $K_{\rho(J)}$ in soundVE. Finally, if the relying party does not know that the judge does not know that the signer uses his signing key only for protocol P, the relying party has no way of knowing whether or not his evidence is sufficient to convince the judge. This is why $K_{\rho(R)}K_{\rho(J)}$ is essential as condition of reachVE.

Note that there is no requirement that a key pair is only used for one protocol. As is well-known, it is not a problem to use a key pair for multiple protocols, as long as these protocols are proved to be compatible together. In practice, any two protocols can be made compatible by adding a protocol identifier to the messages. However, even if all protocols used in practice were compatible, a non-repudiation protocol can still be attacked: the protocol that attacks the non-repudiation protocol is not required to be a protocol used in practice, but can be invented by the attacker specifically for this purpose.

An additional problem is that when getting a certificate that links the identity, key and protocol, the CA needs itself NRI. To understand this, remember that the judge needs to know that agent a in role S in protocol P uses key k only for protocol P. If the CA does not require NRO, a can get a certificate that links key k and protocol P, and later deny that he intentionally did so, preventing the judge from knowing that a uses k only for protocol P. This problem cannot be solved by requiring that the CA checks the identity of the agent that registers the key, as this agent can still claim not having intended to register the key.

Obviously, this gives rise to a bootstrapping problem: an agent should register his key before he can prove non-repudiation of intention with protocol P, and needs to prove non-repudiation of intention before he can register his key. There are several ways to circumvent this problem. A first option is to require that the CA used for generating the certificate does not only generate the certificate, but also generates the key pair itself. This prevents the attack, as it becomes impossible to

register the key at two different CAs for two different protocols. However, the CA will get to know the private key, which is often undesirable, as it requires trust of the signer in the CA. A second option is to require that the key is registered in the physical world, for example at a notary. In this way, the agent that registers his key intended doing so (and cannot deny this intention, assuming the notary is trusted).

As a third possibility, the key could be registered by means of a *digital recording* [Mau04]: any type of projection of physical reality, e.g., a digital image, video or sound recording. Such a recording is evaluated by converting it back to physical form (e.g., an image on a computer screen) and then having it interpreted by humans. This also guarantees that the agent signs intentionally.

5.5 Case Studies

The examples given in the previous sections were all toy examples that perhaps might not be credible. In this section, we look at three more plausible ways in which a virtual multi-protocol could be launched.

5.5.1 Combining Signing and Encryption in PKCS#1v1.5

Existing implementations of non-repudiation protocols are often based on signing algorithms such as PKCS#1v1.5 [Kal98] or PKCS#1v2.1 [PE02], which both implement the RSA algorithm [RSA78]. In this section, we show an attack against the combination of NRI and secrecy in PKCS#1v1.5, obtained by first encrypting and then signing. In the next subsection, we show an attack against NRI obtained by PKCS#1v2.1.

To prevent guessing attacks, PKCS#1v1.5 specifies that to encrypt a message m, the string m', m is encrypted, where m' is a string of random padding. We show that the random padding can be used to create a multi-protocol attack when a message is first encrypted and then signed. Let P be a non-repudiation protocol that specifies that agent S first encrypts a message m for agent R according to PKCS#1v1.5 (i.e., with random padding m'), and then signs it, resulting in the evidence $\{\!|\, \{\!|\, m', m \,|\!\}_{\mathsf{pk}(R)} \,|\!\}_{\mathsf{pk}(S)}$.

Now we specify a new non-repudiation protocol P', in which R sends a nonce n to S, and S replies with $\{\!|\, \{\!|\, m', n \,|\!\}_{\mathsf{pk}(R)} \,|\!\}_{\mathsf{pk}(S)}$. It is easy to check that this guarantees that S cannot deny having sent m' (and that R knows that the message has been sent after he has send n). Note that the encryption does not use padding (which is not a problem, as guessing attacks can be made impossible by choosing the entropy of n high enough).

When a judge is presented with evidence from either P or P', he cannot decide from which of both protocols the evidence originates (it is assumed that m, m', n and S have a fixed, equal length, and that messages cannot be distinguished from random strings - for example, because the message is a decryption key). This gives rise to the following attack trace:

Trace: $a[\text{unrestricted}]; b[P' : \rho = (S \mapsto a, R \mapsto b)]$.

1. $a \to b : m$.
2. $b \to a : \{\!|\, \{\!|\, m', m \,|\!\}_{\mathsf{pk}(b)} \,|\!\}_{\mathsf{pk}(a)}$.

In this attack, agent a pretends to execute P', but sends a message m' instead of the nonce. b assumes that he signs message m' according to P', but a can use the resulting evidence to show that m originated at b according to P.

This attack can also be used to execute a virtual multi-protocol attack. Agent a can execute P to sign m, and in the end deny his intention to sign m, claiming that the evidence results from the execution of P'.

It must be admitted that there are other known problems with the approach of first encrypting and then signing (a relying party can remove the old signature and add his own, claiming authorship of the message [Dav01]). There are also problems known with the approach of first signing and then encrypting. These problems can be solved by either first signing, then encrypting, then signing [Dav01]. However, the (virtual) multi-protocol attack as discussed in this section still applies to these approaches.

5.5.2 RSASSA-PSS and Intention

In order to be provably secure, modern electronic signature schemes are implemented probabilistically rather than deterministically, by incorporating a randomly generated salt value. An example of such a scheme is is the RSA probabilistic signature scheme (RSASSA-PSS) [BR96]. We will see that signing over salt values, however, makes it easier to construct a credible chosen protocol with which a virtual multi-protocol attack becomes possible.

An RSASSA-PSS signature offering NRO by an agent executing role S of message m signed for an agent executing role R has the following form:

$$\{\!|\, ((p_2, s) \oplus \mathrm{MGF}(N)), N, bc \,|\!\}_{\mathsf{pk}(S)} \text{ where } N = \mathsf{h}(p_1, \mathsf{h}(m), s).$$

Here, p_1, p_2 and bc are fixed padding, MGF is a pseudo random function called *message generation function*, s is a random salt value, and $\oplus$ is exclusive or. The final message counts as NRO of message m. The chosen protocol that we will construct is based on RSASSA-PSS, except that we replace $\mathsf{h}(m)$ by a nonce n chosen by the relying party, and the salt s by a new message m', of which we prove NRO. This leads to the following protocol, where the final message counts as evidence of NRO of m'.

Protocol 5.7

1. $R \to S : n$
2. $S \to R : \{\!|\, ((p_2, m') \oplus \mathrm{MGF}(N)), N, bc \,|\!\}_{\mathsf{pk}(S)}$ where $N = \mathsf{h}(p_1, n, m')$

If the relying party is played by an unrestricted agent that sends a message $h(m)$ instead of nonce n, we obtain the following trace:

Trace: $a[P' : \rho(S) = a, \rho(R) = b]; b[\text{unrestricted}]$

1. $b \to a : \mathsf{h}(m)$.
2. $a \to b : \{\!|\, ((p_2, m') \oplus \mathrm{MGF}(N)), N, bc \,|\!\}_{\mathsf{pk}(a)}$ where $N = \mathsf{h}(p_1, \mathsf{h}(m), m')$.

The final message can be either interpreted as NRO evidence of m in RSASSA-PSS, or as NRO evidence of m' in Protocol 5.7. This means that an agent can obtain an NRO evidence of any message in RSASSA-PSS if he can lure another agent into executing Protocol 5.7. Moreover, a virtual multi-protocol attack is possible: an agent can first sign evidence of origin of m by RSASSA-PSS, and later deny that he signed m, claiming that the evidence resulted from signing M' in Protocol 5.7.

RSASSA-PSS can be used as signing algorithm in S/MIME [Ram99], which offers NRO of e-mail messages. S/MIME uses X.509 certificates, which can also be used for other protocols. It is possible to register the *type* of protocol that the key will be used for, e.g., non-repudiation, but not the exact protocol. This is insufficient, as the chosen protocol in this example is also a non-repudiation protocol. This result implies that S/MIME with RSASSA-PSS as signing algorithm cannot be used for contract signing, unless the signing key is registered specifically for S/MIME with RSASSA-PSS.

It should be noted that PGP [Gar95], an alternative scheme offering non-repudiation, is not affected. PGP keys are registered on a PGP key server, which is only used to register keys that will be used for the PGP protocol. Registering a key on a PGP server therefore implicitly confirms that the owner will use the key for (only) for PGP.

5.5.3 Fair Exchange

As a third case study, we demonstrate a credible chosen protocol that invalidates an existing non-repudiation protocol, the ZDB protocol [Zho10]. ZDB offers fair exchange [BOGMR90] of non-repudiation, i.e., when S sends a message m to R, S receives evidence of NRR if and only if R receives evidence of NRO. Fairness cannot be ensured without at least one external agent that is trusted by both parties, called the trusted third party (TTP) [EY80]. ZDB used a trusted third party T which is only involved in exceptional cases. Agent S first sends a message containing ciphertext $c = \{\!|\, k \,|\!\}_m$, awaits evidence of origin of c, sends the (symmetric) key k, and awaits evidence of receipt of k. The following messages are used in the protocol:

- k: message key defined by S,
- $c = \{\!|\, k \,|\!\}_m$: commitment (cipher text) for message m.
- $L = \mathsf{h}(S, R, T, m, k)$: a unique label linking c and k.
- $f_i (1 \leq i \leq 8)$: constant flags indicating the intended purpose of a signed message.
- $\mathsf{EOO}_c = \{\!|\, f_1, R, L, c \,|\!\}_{\mathsf{pk}(S)}$: evidence of origin of c.
- $\mathsf{EOR}_c = \{\!|\, f_2, S, L, c, \{\!|\, \mathsf{pk}(T) \,|\!\}_k \,|\!\}_{\mathsf{pk}(R)}$: evidence of receipt of c.

- $\mathsf{EOO}_k = \{\!|\, f_3, R, L, k \,|\!\}_{\mathsf{pk}(S)}$: evidence of origin of k.
- $\mathsf{EOR}_k = \{\!|\, f_4, S, L, k \,|\!\}_{\mathsf{pk}(R)}$: evidence of receipt of k.
- $\mathsf{sub}_k = \{\!|\, f_5, R, L, k, \mathsf{h}(c) \,|\!\}_{\mathsf{pk}(S)}$: evidence of submission of k to the TTP.
- $\mathsf{con}_k = \{\!|\, f_6, S, R, L, k \,|\!\}_{\mathsf{pk}(T)}$: evidence of confirmation of k issued by the TTP.
- $\mathsf{abort} = \{\!|\, f_8 S, R, L \,|\!\}_{\mathsf{pk}(T)}$: evidence of abortion of a transaction issued by T.

In the original protocol, EOR_c contains $\mathsf{h}(c)$ instead of c, but this is just an optimization [Zho10].

We only give the main protocol, which is executed if no abnormalities occur. For the full protocol, the reader is referred to [Zho10]. In the protocol, $L = \mathsf{h}(S, R, T, m, k)$ is a unique label for a protocol execution, $f_i (1 \leq i \leq 5)$ are constant flags indicating the intended purpose of a message, and sub_k is a message signed by S that R can use to start the recovery protocol with the TTP in the abnormal case where a message does not arrive timely.

If an abnormal case occurs, either both or none of the agents can obtain a message con_k from T. Message EOR_c together with either EOR_k or con_k counts as NRR evidence, while EOO_c with either EOO_k or con_k counts as NRO evidence. For the sake of simplicity, we assume that all parameters have a standard length, and that pairing of messages is implemented by concatenation. This assumption is not vital, as for more complex cases, it is still possible to construct a protocol that attacks the non-repudiation protocol (although more complex).

There are three sub-protocols: Exchange, Abort, and Resolve. In the Resolve sub-protocol, the initiator U is either a or b.

Exchange:

Protocol 5.8

1. $a \to b : b, L, c, T, \{\!|\, \mathsf{pk}(T) \,|\!\}_k, \mathsf{EOO}_c, \mathsf{sub}_k$.
2. If b gives up Then quit.
3. Else $b \to a : a, L, \mathsf{EOR}_c$.
4. If a gives up Then Abort.
5. Else $a \to b : b, L, k, \mathsf{EOO}_k$.
6. If b gives up Then Resolve.
7. Else $b \to a : a, L, \mathsf{EOR}_k$.
8. If a gives up, Then Resolve.
9. Else done.

ABORT:

Protocol 5.9

1. $A \rightarrow T : b, L, \{\!|\, f_7, b, L \,|\!\}_{\mathsf{pk}(a)}$.
2. IF exchange resolved THEN $a \leftarrow T : a, b, L, k, \mathsf{con}_k, \mathsf{EOR}_c$.
3. ELSE $a \leftarrow T : a, b, L, \mathsf{abort}$.

RESOLVE:

Protocol 5.10

1. $U \rightarrow T : a, b, L, \{\!|\, \mathsf{pk}(T) \,|\!\}_k, \mathsf{sub}_k, \mathsf{EOR}_c$.
2. IF exchange aborted THEN $T \leftarrow U : a, b, L, \mathsf{abort}$.
3. ELSE $U \leftarrow T : a, b, L, k, \mathsf{con}_k, \mathsf{EOR}_c$.

The following two authentication protocols together can be used as chosen protocols in a multi-protocol attack. Note that flags are only required to be unique within a protocol, not across protocols. Both of these protocols correctly guarantee that recent aliveness of P to Q. Messages n_1 and n_2 are nonces.

Protocol 5.11

1. $P \rightarrow Q : f_2, n_1, n_2$.
2. $Q \rightarrow P : \{\!|\, f_1, Q, n_1, n_2 \,|\!\}_{\mathsf{pk}(Q)}$.

Protocol 5.12

1. $P \rightarrow Q : f_4, n_1, n_2$.
2. $Q \rightarrow P : \{\!|\, f_3, Q, n_1, n_2 \,|\!\}_{\mathsf{pk}(Q)}$.

This leads to a multi-protocol attack demonstrated in the following attack trace: **Trace:** $a[\text{unrestricted}]; b[Protocol\ 5.11 : \rho(P) = a, \rho(Q) = b], a[Protocol\ 5.12 : \rho(P) = a, \rho(Q) = b]$

1. $a \rightarrow b : f_2, L, c$.
2. $b \rightarrow a : \{\!|\, f_1, a, L, c \,|\!\}_{\mathsf{pk}(b)}$.
3. $a \rightarrow b : f_4, L, k$.
4. $b \rightarrow a : \{\!|\, f_3, a, L, k \,|\!\}_{\mathsf{pk}(b)}$.

Agent b then possesses $(\{\!|\, f_1, a, L, c \,|\!\}_{\mathsf{pk}(b)}, \{\!|\, f_3, a, L, k \,|\!\}_{\mathsf{pk}(a)}) = (\mathsf{EOO}_c, \mathsf{EOO}_k)$, so b can claim NRO of m. This seems undesirable, as a is not aware of sending message m, as she assumed that L, c and k were just nonces. On the other hand, agent a can execute the non-repudiation protocol normally, and afterwards deny that she executed it, claiming that b obtained NRO by executing the authentication protocol.

Now we define a new non-repudiation protocol ZDB', which functions as a chosen protocol that invalidates ZDB. ZDB' is equal to ZDB, except that flags f_1 and f_2 are swapped, and that c and L are swapped in EOR_c, i.e.,

$$\mathsf{EOR}_c = \{\!|\, f_1, S, L, \{\!|\, \mathsf{pk}(T) \,|\!\}_k, c \,|\!\}_{\mathsf{pk}(R)}.$$

Furthermore, L and $\{\!|\, \mathsf{pk}(T) \,|\!\}_K$ are half the length of the other parameters. This clearly does not change the correctness of the protocol. However, the following attack trace becomes possible, in which $L = (L_1, L_2)$ and s is any string.

Trace: $a[ZDB' : \rho(S) = a, \rho(R) = b); b[\text{unrestricted}]$

1. $b \to a : a, L_1, c, T, L_2, \{\!|\, f_2, a, L_1, c \,|\!\}_{\mathsf{pk}(b)}, \{\!|\, s \,|\!\}_{\mathsf{pk}(b)}$.
2. $a \to b : b, L, \mathsf{EOR}'_c$.

Note that a has no way to find out that the parameters L, c,$\{\!|\, \mathsf{pk}(T) \,|\!\}_k$ and sub_k are incorrect, as she does not yet know k and m, and that EOO_c matches the expected value. After this trace is executed, b possesses the message $\mathsf{EOR}'_c = \{\!|\, f_1, b, L_1, L_2, c \,|\!\}_{\mathsf{pk}(a)} = \{\!|\, f_1, b, L, c \,|\!\}_{\mathsf{pk}(a)}$, which is equal to the evidence EOO_c in ZDB.

Analogously, b can obtain EOO_K from a (with a second chosen protocol). As a can obtain both EOO_c and EOO_k, a can obtain evidence of origin without b having originated m. This allows for a multi-protocol attack if R considers it possible that S executes ZDB' (with the relevant key): S can deny having originated m (even if he did originate it), by claiming that his evidence results from the execution of another protocol.

Note that to create the protocols that invalidate the claims of ZDB, it was sufficient to change the argument order and swap two flags. This indicates that even a slight change in a protocol, for example a different version, can be used to invalidate the original protocol.

5.6 Related Work

Formalization of non-repudiation starts with Zhou and Gollmann [ZG98], who formally express (versions of) non-repudiation as goals in SVO logic. For example, NRO is defined as 'J believes that (a said M)'. This corresponds to our soundGE property. The other properties are not considered. Analysis of non-repudiation in terms of evidence dates back to Schneider [Sch98], who uses Communicating Sequential Processes to model non-repudiation protocols. He requires that if pieces of evidence are sent from an agent to the judge, then another agent must have sent (in the case of NRO) or received (in the case of NRR), a particular message. This loosely corresponds to our soundGE and soundVE properties. However, in the analysis of Schneider, it is not taken into account that either the sender or the judge might not be able to inspect the evidences (for example, because the evidence is encrypted), and that the properties therefore are not equivalent. He also does not consider the reachVE property. Moreover, both Zhou & Gollmann and Schneider do not make the assumptions that are made about the protocol

participants non-repudiation explicit. Admittedly, in the protocols considered in [ZG98, Sch98], these additional properties are satisfied trivially. Bella & Paulson [BP01] have used the theorem prover Isabelle to verify non-repudiation protocols. They also do not distinguish between soundGE and soundVE. Klay & Vigneron [KV08] formalize non-repudiation as an authentication problem, i.e., they require that the relying party needs to authenticate the signer and the TTP. We have seen that this only partially models non-repudiation, as authentication does not model the aspect that the relying party needs to convince the judge.

The notion of non-repudiation is closely related to the notion of *accountability*, which says that a judge blames protocol participants if and only if they misbehave. Various formal definitions of accountability have been proposed, i.e., [KTV10, KR10]. Much attention has also been paid to the formalization of the fair exchange property [BOGMR90], rather than the non-repudiation property itself.

Another property often considered in the context of contract-signing is *abuse-freeness* [GM99], meaning that at no point a participant can prove to others that he is capable of choosing whether to validate or invalidate a contract.

5.7 Conclusion

It is well-known that agents need to restrict the use of their cryptographic keys to a limited set of protocols if multiple security protocols are executed in parallel. We have shown that this restriction is not sufficient, as it allows for virtual multi-protocol attacks, i.e., situations in which a security property holds, but is not known to hold by some agent. To prevent such attacks, we proposed an additional assumption, which states that agents know for which protocols other agents use their keys. This assumption can be satisfied by making the public key infrastructure not only link keys to agents, but also to the protocol for which the key is used. Such an assumption is currently not observed in practice in protocols claiming to offer non-repudiation and contract signing. For instance S/MIME [Ram99], a protocol of which the documentation explicitly states that it offers non-repudiation, uses generic keys and certificates that allow keys to be used in other protocols as well. Therefore, an agent can deny his signature by claiming that it resulted from the execution of another protocol. This implies that S/MIME is only suitable for non-repudiation if agents know that the other agents do not use their keys for different purposes.

Virtual multi-protocol attacks can occur in other types of protocols as well. For instance, in a secrecy protocol P, if the receiver does not know that the signer executes only protocol P, then the signer can apply a virtual multi-protocol attack by claiming that he did not execute protocol P, and that the claim is broken, i.e., that the message is not secret anymore. Our results also extend to intentional authentication. For example, when an agent transfers money from his bank account, the bank should require that the agent intends to do so. This can be accomplished by letting the agent sign his command. Our results indicate that this is not sufficient in general: the bank should also require that the agent registers his signing key for the protocol that the bank uses.

However, virtual multi-protocol attacks are especially relevant for non-repudiation

protocols for three reasons. First, in non-repudiation protocols, it is in the advantage of the signer to state that the security claim does not hold, as, for instance, being able to deny a contract might be profitable. Secondly, non-repudiation protocols are executed more often in open systems where the participants do not have knowledge about each other. Thirdly, in non-repudiation protocols, there is an external judge involved, who also must have knowledge about the other agents.

Moreover, we have given a new definition of non-repudiation. This definition is an improvement on previous definitions, as it excludes some intuitively incorrect protocols that are accepted by previous definitions.

Some non-repudiation protocols make use of a trusted third party (TTP). We expect that our analysis can be easily extended to incorporate this. A formal analysis of this situation is left for future work.

6

Incentives in Security Protocols

Abstract. We propose a methodological framework for analyzing interaction protocols that takes into account the incentives of agents. The framework allows for a more fine-grained analysis of such protocols. We formally define correctness of a protocol given a notion of rationality, and possibly given the utilities of the agents and a set of agents that supports the objective of the protocol. Finally, we describe the security level of a protocol as the minimal sets of participants supporting the objective of the protocol that are necessary to achieve this objective.

6.1 Introduction

From a game-theoretic perspective, security protocols are an interesting class of games since they have a *goal*, i.e., an outcome that is preferred by the designer of the protocol. In security protocols, some participants have a strong incentive to deviate from the protocol. Security protocols enforce certain goals, provided that at least some of the participants follow the original protocol.

When studying the interaction between security protocols and game theory, researchers have usually considered protocol execution as a game with the very pessimistic assumption that the only goal of the other participants ("attackers") is to break the intended security property of the protocol. In other words, the classical requirement states that a protocol is correct if the agents supporting the objective have a strategy such that for all strategies of the other agents, the objective of the protocol is satisfied. Game-based analysis in such a way has been performed in [KR02], among others. Recently, protocols have been analyzed with respect to game theory's notions of *rationality* [Mic10, ACH11], where the preferences of the participants are taken into account.

In this chapter, we point out that the above definition of correctness can be too strong, because violation of the objective may be achievable only by irrational responses from the other agents. On the one hand, the only behavior violating the security constraints might never going to be chosen by rational attackers. On the other hand, the definition may also prove too weak when the objective of the protocol can be only achieved by an *irrational* joint strategy of agents supporting the objective, in other words: one that they will never choose to play. In this chapter, we show that this classical requirement is neither necessary nor sufficient if we have any – even very weak – idea of what drives the participating parties.

To describe and predict rational behavior of agents, game theory has proposed a number of *solution concepts* [OR94, SLB09]. Each solution concept represents a different notion of rationality and ability of coordination, which may be more or

less applicable in different contexts.

Another strategic issue that is central in the study of many games is the question of how much (if at all) participants are able to *coordinate* their strategies. In the study of security protocols, it is important that if a coalition has a strategy to achieve some objective (such as signing a contract), each of the coalition's members must be able to *identify* their actions to be followed in the strategy. The question of how much coordination is possible between different participants is closely related to the question which solution concept to apply. Hence, choosing an adequate solution concept is a key step in performing a meaningful protocol analysis.

For example, Nash equilibrium, which is used in all rational analysis of security protocols that we are aware of, assumes coordination capabilities that participants may not have in a concrete protocol setting. We do not fix any particular solution concept, and instead take it to be a parameter of the problem.

Our approach applies to all protocols where the objective (e.g., secrecy, authentication) is well-defined for individual runs. As running example, in this chapter we focus on contract signing protocols as an example of fair-exchange protocols. In contract-signing protocols, Alice and Bob want to exchange digitally signed contracts [BOGMR90, ASW98, GJM99]. We abstract away from the actual implementation of the protocols, and model them as game frames where strategy profiles correspond to possible ways of executing the protocol. Moreover, utility functions represent agents' preferences over different possible outcomes of the interaction.

Consider Alice's point of view. She knows that if the protocol run is supposed to be effective, Bob must cooperate in some way. Hence Alice can assume, at the very least, that Bob will play a "nice" strategy that does not completely preclude the exchange of contracts. Therefore, one reasonable starting point would be to require that Alice has a strategy that ensures effectivity as long as Bob plays any strategy that is "nice" in this sense.

More generally, Alice's reliance on Bob playing "nice" can be captured game-theoretically as Alice being able to reason about Bob's incentives. With solution concepts, game theory provides useful tools to formalize situations like these and analyze their strategic properties.

The aim of this chapter is to provide a methodological framework that uses game-theoretic models of agents' incentives to obtain a finer-grained analysis of properties of interaction protocols. First, we propose that a protocol satisfies its objective if the objective is satisfied by all *rational* behaviors of participants. To define rational behavior, we need an accurate model of agents' incentives. Since the exact incentives of the agents are often unknown, we consider the following alternative: We assume a subset of agents to be in favor of the security (and/or functionality) properties in question. For instance, a bank should not favor fraud that decreases its revenue, a contract signing party should not prefer to be cheated by the other party, etc. Analogously to the classic definition of security (a protocol should achieve its objective as long as a specific group of participants follows the protocol), we give a very natural definition in the game-theoretic setting: Every rational protocol run must satisfy the protocol goal, as long as a specific group of participants are in favor of the protocol goal. The group defines the *game-theoretic security level* of the protocol.

Our main contributions are the following:

1. We define rational correctness notion for security protocols,
2. we give an appealing characterization about security properties that can be defended in the above sense in the case of using Nash equilibria,

It is important to note that we do not fix a single notion of rationality here, but consider it as a parameter to our model.

6.2 Security Protocols

We begin by defining the models of security protocols that we use in this thesis. A protocol is a formal specification of how agents should interact. Protocols might contain *choice points* in which more than one action is available to the agents. If a security protocol contains *choice points* in which more than one action is available to the agents, the protocol is *non-deterministic*. A *strategy* of an agent in a protocol is a conditional plan, i.e., a function that specifies for each choice point which action to take. A set of deterministic strategies, one for each agent, uniquely determines a *run* of the protocol, i.e., the sequence of actions that the agents take. We say that an agent is *honest* if he follows the protocol specification, and *dishonest* otherwise, i.e., when he behaves in a way that is not allowed by the protocol. In that case, the agent is only restricted by the physical and logical actions that are available in the environment. For instance, in a security protocol, agents can do anything that satisfies properties of the cryptographic primitives, assuming perfect cryptography (as in [KR03]).

In this chapter, we mainly focus on two-party contract signing protocols. In such protocols, two agents, which we will call Alice and Bob, intend to sign a contract. Many security properties of protocols can be expressed in terms of desirable properties, or *objectives*, of runs. The two main objectives of runs in contract signing protocols are *fairness* and *effectivity*.

Remember that fairness should be guaranteed for honest agents even if the other agents are not honest. In order to achieve fairness, contract signing protocols have to make use of a *trusted third party* T [PG99], which is assumed to be honest. A protocol run is *effective* if at the end of the run, both agents have the signature of the other agent. Obviously, if a run is effective, it is also fair.

Recall that in the traditional definition of effectivity of a contract-signing protocol, it is required that the agents together must be able to enforce an effective run, while any of the agents should be able to enforce fairness for himself.

In this chapter, we do not study the cryptographic aspects of security of contract-signing, but instead consider the question whether a protocol allows agents that play "rationally" to reach the relevant security goals. Therefore, we take a high-level view of the protocol and assume that attacks on the protocol in the cryptographic sense (i.e., forging of messages, multi-session attacks, type flaw attacks, etc.) are prevented by the usual cryptographic tools (signing, encryption, typing of messages, etc.), and that all messages that are being transmitted are labeled with

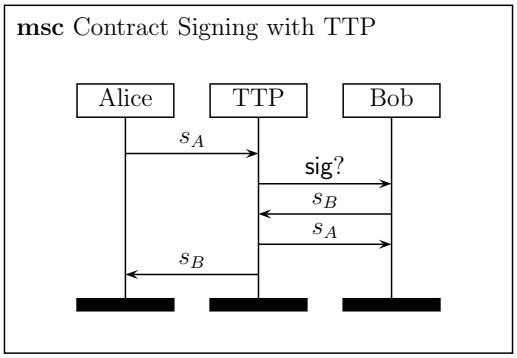


Figure 6.1: A contract-signing protocol

the type of the message, the name of the sender, the name of the intended recipient, the name of the TTP which is agreed upon, and an identifier linking the message to the protocol session. Similarly, we assume that the communication channels are *resilient*, i.e., that all messages are delivered after a finite, but unknown amount of time.

We will usually assume that Alice prefers the situation where both Alice and Bob receive a signed contract over the one where neither of them do. Moreover, her most preferred option is to receive a signed contract while Bob does not receive one, and her least preferred outcome is to not receive a contract in spite of Bob receiving one.

Example 6.1. *Figure 6.1 displays a simple contract signing protocol [BOGMR90], where Alice and Bob exchange signatures. It makes use of a* trusted third party *(TTP) T, which is assumed to be honest. The protocol specifies that first Alice sends her signature s_A to the TTP, and then the TTP requests Bob's signature with the message* sig?*. Subsequently, Bob sends his signature s_B to the TTP. Finally, the TTP forwards the signatures to Bob and Alice.*

Clearly, in a real-world setting, each participant in a protocol can stop execution of the protocol at any point. To model this, we formally consider protocols as non-deterministic.

Moreover, we assume that each participant has the choice to stop execution of the protocol at any point. In this protocol, fairness is intuitively guaranteed, because the TTP is honest and will not send out s_A and s_B before he has collected both signatures.

6.2.1 Rationality and Coordination in Security Protocols

It is traditionally required that a correct protocol must satisfy the relevant properties in all possible runs [CR10, KR03]. In case of contract signing that implies a.o. that all combinations of honest strategies are fair. We claim that this requirement is often too strong. Consider, for example, the ASW contract signing protocol [ASW98] (see also Section 6.5.1) where agents have the right to abort or stop the execution of the protocol. When a participant is "cheated", he can always call the "resolve" protocol. If he does not do this, the protocol run is not fair according to the above definition. This seems intuitively wrong in a situation

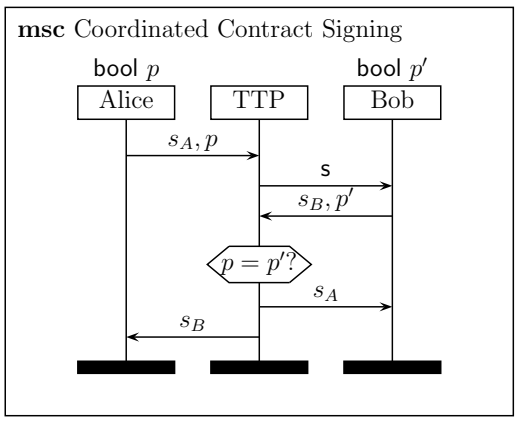


Figure 6.2: A contract-signing protocol in which coordination is required

where the participant voluntarily gave up enforcing fairness, presumably because he pursued other goals. In particular, we would like to distinguish between situations where an agent is deprived of fairness, and ones where he *prefers* not to obtain it. This calls for a model of *reasons* for choices of the participants in a protocol.

Moreover, effectiveness is usually deemed to hold if there exists a run that is effective [CR10]. We claim that this is too weak. First, the participants might not be able to coordinate on playing a successful run. Second, even if they can do that, they might still not be willing to, as it comes against other goals that they pursue.

In fact, typical analysis of contract signing is also based on an implicit model of incentives. One usually assumes that a participant, say Alice, has an incentive to obtain a fair run for her but not for the others (Bob in this case), as Alice prefers obtaining the signature of Bob without sending a signature herself. Contract signing protocols are supposed to prevent such attacks, and that is where a trusted third party is needed. We point out that both assumptions can be wrong: that Alice wants Bob's signature as well as that she wants to deprive Bob of her signature.

The following protocol serves as an example where the agents need to coordinate in order to exchange signatures. This demonstrates that the requirement that there exists a successful run is insufficient.

Example 6.2. *Consider the protocol in Figure 6.2, which is similar to the protocol in Figure 6.1, except that the agents have to send Boolean values p and p′ to the TTP. The TTP only forwards the signatures if both agents have sent the same Boolean. As the agents need to choose the same Boolean, coordination is required in order to exchange signatures.*

The above protocol is clearly contrived, however such concerns more realistically appear in situations when Alice and Bob use a contract signing protocol in which the TTP has limited memory, and thus cannot keep every contract received in memory, and hence there is an upper bound on the maximal number of open sessions. To achieve contract signing, Alice and Bob thus have to coordinate the time of contacting the TTP.

(Note that coordinating session numbers can be avoided, since it can be assumed

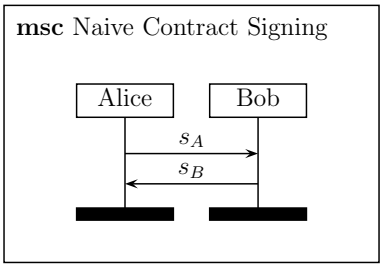


Figure 6.3: A naive contract-signing protocol

that the TTP can uniquely identify the session when given the contract text and the public keys of Alice and Bob.)

The following example demonstrates a protocol where the objective (fairness) can be in principle satisfied, but only if the participants choose irrational behavior.

Example 6.3. *Figure 6.3 depicts a naive contract signing protocol. In this protocol, Alice sends a signature to Bob, who responds by sending back his signature. Again, we assume that Alice and Bob can choose to stop executing the protocol at any moment. The run where Bob and Alice both send their signatures is fair. However, it is irrational for Alice to send her signature, as Bob has an incentive to stop the protocol (before sending his signature) instead of sign, and thereby causing harm to Alice.*

We will use game models to represent incentives and study their impact on correctness of protocols.

6.2.2 Game-theoretic Models of Interaction

We will use *normal-form games* to provide an abstract representation of agents' possible strategies, runs resulting from playing the strategies, and preferences of agents with respect to runs.

Definition 6.1 (Frames and games)**.** *A* (normal-form) game frame *is a tuple* (N, Σ, Ω, o)*, where* $N = \{A, B, \ldots\}$ *is a finite set of* agents*,* $\Sigma = \Sigma_A \times \Sigma_B \times \ldots$ *is a set of strategy profiles (*Σ_i *is a strategy for each* $i \in N$*),* Ω *is the set of outcomes, and* $o : \Sigma \to \Omega$ *is a function mapping each strategy profile to an outcome. We write* s_{-i} *for the tuple of strategies by agents* $N \backslash \{i\}$*, and if* $N = \{A_1, \ldots, A_n\}$*, we write* $(s'_{A_i}, s_{A_{-i}})$ *for the strategy profile* $(s_{A_1}, \ldots, s_{A_{i-1}}, s'_{A_i}, s_{A_{i+1}}, \ldots s_{A_n})$*.*

It is often assumed that $\Omega = \Sigma$*, in which case we will write* (N, Σ) *instead of* $(N, \Sigma, \Sigma, \mathrm{id})$*, where* id *is the identity function.*

We will use both notations interchangeably.

A normal-form game (F, u) *consists of a game frame* F *together with a* utility profile $u = \{u_A, u_B, \ldots\}$ *where* $u_i : \Sigma \to \mathbb{R}$ *is a* utility function *assigning utility values to strategy profiles. We write* $\mathcal{G}$ *for the class of all normal-form games.*

We assume that agents play rationally when executing the protocol. Game theory uses *solution concepts* to define which strategy profiles capture rational interactions

	X	Y
X	$1, 1$	$5, 0$
Y	$0, 1$	$2, 2$

Figure 6.4: Example normal-form game

(given a utility profile). Formally, a solution concept is a function $SC : \mathcal{G} \to \mathcal{P}(\Sigma)$ that given a game returns a set of *rational* strategy profiles. Well-known solution concepts include e.g. Nash equilibrium (NE), dominant and undominated strategies, Stackelberg equilibrium, Pareto optimality etc. For definitions of various solution concepts and detailed discussion, we refer to the textbooks [OR94, SLB09].

As the appropriate notion of rationality may depend both on the protocol and on the situation in which it is analyzed, we do not fix a single solution concept but instead treat it as a parameter of our analysis.

Definition 6.2. *A solution concept is a function* $SC : \mathcal{G} \to \mathcal{P}(\Sigma)$.

We proceed by defining the solution concepts we use in this chapter, given a game $(N, \Sigma), u)$.

Nash equilibrium A strategy profile $s = (s_A, s_B, \ldots)$ is a *Nash equilibrium*, written $s \in NE(G)$, if no agent can unilaterally improve, i.e., if there does not exist an agent $i \in N$ with a strategy s_i' such that $u_i(s_i', s_{-i}) > u_i(s_i, s_{-i})$.

Equilibrium in undominated strategies A strategy s_i is *undominated* if no other strategy is at least as good against all strategies of the other agents and strictly better against some strategy the other agents, i.e., if there is no s_i' such that $u_i(s_i', s_{-i}) > u_i(s_i, s_{-i})$ for some strategies s_{-i} and $u_i(s_i', s_{-i}) \geq u_i(s_i, s_{-i})$ for all strategies s_{-i}. A strategy profile $s = (s_A, s_B, \ldots)$ is a *equilibrium in undominated strategies*, written $s \in Undom(G)$, if s_i is undominated for every $i \in N$.

Maximal sum A strategy profile s is *max-sum* if the sum of the utilities is maximal, i.e., if there exists no strategy profile s' such that $\sum_{i \in N} u_i(s') > \sum_{i \in N} u_i(s)$.

Note that the above-mentioned solution concepts do not only cover different notions of "goodwill" (maximal sum assumes that agents have an interest in maximizing each other's utility, while Nash equilibrium does not), but also different abilities to coordinate. For example, finding a Nash equilibrium requires communication between the agents if there is more than one equilibrium in the game. It also assumes that each agent has some knowledge about the other agents' utility functions. On the other hand, each agent can identify an undominated strategy without requiring either coordination or information about the other agent's incentives.

Example 6.4. *Consider a normal-form game* $G = (N, \Sigma, u)$ *with* $N = \{A, B\}$, $\Sigma_A = \Sigma_B = \{X, Y\}$ *and* u *as depicted in Figure 6.4. We have* $NE(G) = \{(1, 1)\}$, $Undom(G) = \{(1, 1), (5, 0)\}$, *and* $MS(G) = \{(5, 0)\}$.

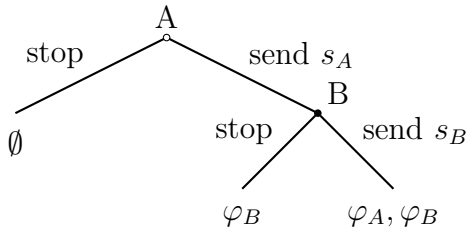


Figure 6.5: The protocol from Figure 6.3 expressed as an extensive-form game frame

6.2.3 Protocols as Games

In this section we define our game models of protocols, which will provide the basis for the analysis of protocol correctness in Section 6.3. First, we define *extensive-form game frames*, that is, multi-step games represented by the tree of possible runs without incentives indicated, as they are more intuitive when modeling protocols than normal-form games. Then, we convert these into *normal-form game frames*, i.e., normal-form games without incentives. Finally we show how we can obtain normal-form games, by adding the incentives of the agents to the model.

6.2.4 Security Protocols as Game Trees

Definition 6.3. *An* extensive-form game frame *is a tuple* (N, H, P, Ω, o)*, where:*

- $N = \{A, B, \ldots\}$ *is the set of* agents.
- H *is a set of sequences (finite or infinite) that satisfies the following three properties: 1) the empty sequence* $\emptyset$ *is a member of* H*; 2) if* $(a^1, \ldots, a^j) \in H$ *(where* j *may be infinite) and* $k < j$ *then* $(a^1, \ldots, a^k) \in H$*; 3) if an infinite sequence* $a = (a^1, \ldots)$ *satisfies* $(a^1, \ldots, a^k) \in H$ *for every positive integer* k *then* $a \in H$*. (Each member of* H *is a history; each component of a history is an action taken by an agent.) A history* $(a^1, \ldots, a^k) \in H$ *is* terminal *if it is infinite or if there is no* a^{k+1} *such that* $(a^1, \ldots, a^k, a^{k+1}) \in H$*. We write* Z *for the set of terminal histories.*
- P *is a function that assigns to each nonterminal history* $h \in H \backslash Z$ *a member of* N*, being the agent who takes an action after the history* h*.*
- Ω *is the set of* outcomes*.*
- $o : Z \to \Omega$ *is a function mapping each terminal history to an outcome.*

With an extensive-form game frame (N, H, P, Ω, o)*, for each agent* $i \in N$ *we associate a strategy* Σ_i*, which is a function that assigns an* action *a to each nonterminal history* $h \in H \backslash Z$ *for which* $P(h) = i$ *such that* $(h, a) \in H$*. If* Σ_i *is a set of strategies for each* $i \in N$*, we call* $\Sigma = \Sigma_A \times \Sigma_B \ldots$ *the set of* strategy profiles*. For each strategy profile* $s \in \Sigma$ *in the extensive-form game* (N, H, P, u)*,* s *yields a* run $run(s)$*, which is defined as the (possibly infinite) history* $(a_1, \ldots) \in H$ *such that for relevant* $i \geq 0$*, we have* $s_{P(a_1,\ldots,a_i)}(a_1, \ldots, a_i) = a_{i+1}$*.*

Example 6.5. *Consider the protocol in Figure 6.2. To denote the outcomes, we use variables φ_i for each $i \in N$, where φ_i means that agent i has received the signature of the other agent. Now we express this protocol as the extensive-form game (N, H, P, Ω, o), where*

- $N = \{A, B\}$,
- $H = \{\emptyset, (stop), (send\ s_A), (send\ s_A, stop), (send\ s_A,\ send\ s_B)\}$,
- $P(\emptyset) = A$, $P(\{send\ s_A\}) = B$,
- $\Omega = \mathcal{P}(\{\varphi_A, \varphi_B\})$,
- $o(stop) = \emptyset$, $o(send\ s_A, stop) = \{\varphi_B\}$, *and* $o(send\ s_A, send\ s_B) = \{\varphi_B, \varphi_B\}$.

This extensive-form game is depicted in Figure 6.5.

We note that some solution concepts, such as subgame perfect Nash equilibria, can be only defined in extensive-form games [OR94]. We leave analysis of protocols based on subgame-perfectness for future work.

6.2.5 Game Frames for Protocols

In order to focus on the strategic interactions of agents in a protocol, we only model the possible strategies that agents have, and the outcomes in which playing these strategies result. Thereby, we abstract away from the internal (temporal) structure of the protocol, and model it as a normal-form game.

We will use normal-form games in the rest of the chapter to focus on the interplay between agents' incentives and the objective of the protocol, rather than the internal structure of the protocol execution.

Every extensive-form game frame can be converted into a normal-form game frame.

Definition 6.4. *The conversion of extensive-form game frame $G = (N, H, P, \Omega, o)$ with strategy profiles Σ to a normal-form game frame is (N, Σ, Ω, o') in which for each agent $i \in N$, $o'_i(s) = o_i(run(\Sigma))$ for all $i \in N$, $s \in \Sigma$.*

For example, the normal-form game frame corresponding to the game tree from Figure 6.5 is depicted in Figure 6.6b.

A *joint strategy* $\sigma_{\mathcal{A}}$ for agents $\mathcal{A}$ is a tuple of strategies, one for each agent $i \in \mathcal{A}$. If σ is a joint strategy for agents $\mathcal{A}$, and σ' is a strategy profile, then σ' is an *extension* of σ, written $\sigma \sqsubset \sigma'$, if for all agents $i \in \mathcal{A}$, $\sigma_i = \sigma'_i$.

Example 6.6. *Consider the protocol in Figure 6.1, and the strategies of Alice and Bob* stop *(meaning stopping before sending the signature) and* ***sign*** *(running the protocol honestly). The protocol can then be modeled as the game frame (N, Σ, Ω, o), with $N = \{A, B, T\}$, $\Sigma_T = \{-\}$ (the trusted third party T is deterministic, i.e., has only one strategy), $\Sigma_A = \Sigma_B = \{\mathsf{stop}, \mathsf{sign}\}$, $\Omega = \mathcal{P}(\{\varphi_A, \varphi_B\})$, $o(\sigma) = \{\varphi_A, \varphi_B\}$ if $\sigma = (\mathsf{sign}, \mathsf{sign}, -)$, and $o(\sigma) = \emptyset$ if $\sigma \neq (\mathsf{sign}, \mathsf{sign}, -)$. The game frame is displayed*

	$Stop$	$Sign$
$Stop$	$\emptyset$	$\emptyset$
$Sign$	$\emptyset$	$\{\varphi_A, \varphi_B\}$

(a)

	$Stop$	$Sign$
$Stop$	$\emptyset$	$\{\varphi_B\}$
$Sign$	$\emptyset$	$\{\varphi_A, \varphi_B\}$

(b)

	$Stop$	$Sign_1$	$Sign_2$
$Stop$	$\emptyset$	$\emptyset$	$\emptyset$
$Sign_1$	$\emptyset$	$\emptyset$	$\{\varphi_A, \varphi_B\}$
$Sign_2$	$\emptyset$	$\{\varphi_A, \varphi_B\}$	$\emptyset$

(c)

	$Stop$	$Sign_1$	$Sign_2$
$Stop$	$\emptyset$	$\emptyset$	$\emptyset$
$Sign_1$	$\emptyset$	$\emptyset$	$\{\varphi_A, \varphi_B\}$
$Sign_2$	$\emptyset$	$\{\varphi_A, \varphi_B\}$	$\{\varphi_A, \varphi_B\}$

(d)

	$Stop$	$Sign_1$	$Sign_2$
$Stop$	$\emptyset$	$\emptyset$	$\emptyset$
$Sign_1$	$\emptyset$	$\emptyset$	$\{\varphi_A, \varphi_B\}$
$Sign_2$	$\emptyset$	$\{\varphi_A, \varphi_B\}$	$\emptyset$
$Sign_3$	$\emptyset$	$\{\varphi_A, \varphi_B\}$	$\{\varphi_A, \varphi_B\}$

(e)

Figure 6.6: Game frames for contract signing

in Figure 6.6a. The protocol from Figure 6.3 can be modeled as the game frame in Figure 6.6b.

The naive contract signing protocol from Figure 6.3 can be modeled in a similar way (Figure 6.6b). The only difference with the protocol in Figure 6.1 is that we define $o(\sigma) = \{\varphi_B\}$ *if* $\sigma \neq \{\mathsf{notsign}, \mathsf{sign}, -\}$.

Finally, the contract signing protocol from Figure 6.2, in which coordination is required, can be represented as the game frame in Figure 6.6c.

6.2.6 Adding Incentives to Protocols

In order to model protocols as games, we need to formalize agents' preferences with respect to outcomes.

Note that we assign utilities to outcomes and not to strategy profiles, as we want to model the fact that if two strategy profiles have the same effect (for example because the same agents get contracts), agents should have equal utilities for these strategy profiles.

Definition 6.5. *Whenever* $P = (N, \Sigma, \Omega, o)$ *is a normal-form game frame and* $u = \{u_A, u_B, \ldots\}$ *is an outcome utility profile, the normal-form game corresponding to* (P, u) *is defined as* $\Gamma(P, u) = (N, \Sigma, (u'_A, u'_B, \ldots))$, *where* $u'_i(s) = u_i(o(s))$ *for all agents* $i \in N$.

Now we show how we can model protocols represented as game frames together with a utility function as normal-form games.

Example 6.7. *Assume the following utility function for A:* $u_A(\{\varphi_A, \varphi_B\}) = 2$; $u_A(\emptyset) = 1$; $u_A(\{\varphi_A\}) = 3$; $u_A(\{\varphi_B\}) = 0$. *The utility function for B is symmetric. Thus, both agents prefer the exchange of signatures over no exchange; moreover, the most preferred option for an agent is to get the signature while the other agent does not, and the least preferred option is not to get the signatures while the other agent does. Combining this utility function with the game frames from Figures 6.6 leads to the games depicted in Figure 6.7.*

Now assume that the agents are in fact not willing to sign. In that case, agents prefer the outcome $\emptyset$ *over* $\{\varphi_A, \varphi_B\}$, *so the utility function of A is as follows:*

	Stop	*Sign*
Stop	1, 1	1, 1
Sign	1, 1	2, 2

(a) G_1

	Stop	*Sign*
Stop	1, 1	0, 3
Sign	1, 1	2, 2

(b) G_2

	Stop	$Sign_1$	$Sign_2$
Stop	1, 1	1, 1	1, 1
$Sign_1$	1, 1	1, 1	2, 2
$Sign_2$	1, 1	2, 2	1, 1

(c)

	Stop	*Sign*
Stop	2, 2	2, 2
Sign	2, 2	1, 1

(d) G_1

	Stop	*Sign*
Stop	2, 2	0, 3
Sign	2, 2	1, 1

(e) G_2

	Stop	$Sign_1$	$Sign_2$
Stop	2, 2	2, 2	2, 2
$Sign_1$	2, 2	2, 2	1, 1
$Sign_2$	2, 2	1, 1	2, 2

(f) G_3

Figure 6.7: Games for contract signing protocols

$u_A(\{\varphi_A, \varphi_B\}) = 1; u_A(\emptyset) = 2; u_A(\{\varphi_A\}) = 3; u_A(\{\varphi_B\}) = 0$. *The utility function for agent B is symmetric. Now, the game frames from Figure 6.6 can be modeled as the normal-form games in Figure 6.7d-f.*

6.2.7 Modeling Security Objectives

A security protocol is designed to achieve one or more *security properties* and/or *functionality properties*. We only consider properties that can be expressed in terms of individual strategy profiles having a certain good property. To model this, we define a set of outcomes that we consider good, called the *objective*.

Note that this does not restrict us to trace properties as in defining which runs are correct, we might use information from the other runs as well.

For example, consider balance in contract signing: A contract signing protocol is *balanced* [CKS01], if there is no reachable state in the protocol execution where one of the signers can decide unilaterally whether the contracts are exchanged or not. This property is desirable e.g., when trading at the stock market. One can define balance by stating that a strategy profile is correct if at no point in the run, an agent has a choice between getting the contract or aborting. Although this is not a trace property, we still can define whether individual runs do or do not satisfy the objective.

Definition 6.6. *Given a game frame* (N, Σ, Ω, o)*, an* objective *is a non-empty set* $\gamma \subseteq \Omega$*.*

Given a game frame (N, Σ, Ω, o), an *objective* is a non-empty set $\gamma \subseteq \Omega$.

Example 6.8. *In a fair strategy profile, either both agents obtain the signature of the other agent, or none of them does, and in an effective strategy profile, both agents obtain the other agent's signature. Hence we write* $\gamma_{\mathsf{fair}} = \{\emptyset, \{\varphi_A, \varphi_B\}\}$ *for the fairness objective, and we write* $\gamma_{\mathsf{eff}} = \{\{\varphi_A, \varphi_B\}\}$ *for the objective of effectiveness.*

6.3 Game-Based Security Analysis

First, we give a definition of correctness of protocols, assuming that the utility profile is known. Then we proceed to the more general situation, where correctness of a protocol is to be verified without knowing the utility profiles of the agents. In this section, we give a new definition of correctness of security protocols that takes into account the incentives of agents, represented as utility functions.

6.3.1 Incentive-Based Correctness

We have seen on the one hand, that defining correctness of a protocol as all strategy profiles satisfying the objective is too strong for our purpose, as it also requires runs to satisfy the objective that agents will never play as they are irrational for them. On the other hand, requiring that some strategy profile satisfies the objective is too weak for our purpose, as agents might be unable to coordinate on such a strategy profile, or this strategy profile might be irrational for them to play. while requiring that some strategy profiles satisfy the objective is too weak. We propose an intermediate definition, which states that a protocol is correct if all *rational* runs satisfy the objective.

We have seen in the previous section that the requirement that all strategy profiles satisfy the objective might be too strong, Instead, we require that all *rational* runs satisfy the objective. In case there are no rational runs, all outcomes are possible, and all of them must satisfy γ. The non-existence of a rational run can occur, for example, if the agents lack the ability to coordinate their strategies.

Formally, we define correctness of a protocol P with respect to a utility profile u, an objective γ, and a solution concept SC. First, we require that any strategy profile in the game corresponding to P and u that is rational according to SC satisfies the objective γ. Second, we require that there exists a strategy profile in the game corresponding to P and u that is rational according to SC.

Definition 6.7. *A protocol represented as game frame $P = (N, \Sigma, \Omega, o)$ with utility profile u is* correct with respect to objective γ under solution concept SC*, written $(P, u) \models_{SC} \gamma$, if and only if:*

$$\begin{array}{ll} SC(P,u) \subseteq \gamma \text{ if } & SC(P,u) \neq \emptyset \\ \gamma = \Omega & \text{else.} \end{array}$$

Protocol verification *is the following decision problem:*

- ***Input:*** *A protocol P, a utility function u, an objective γ and a solution concept SC.*
- ***Question:*** *Does $(P, u) \models_{SC} \gamma$ hold?*

Example 6.9. *Consider game G_2 in Figure 6.7b for the naive contract signing protocol. We saw that if Alice signs, Bob might stop the protocol, resulting in the worst possible utility for Alice. Therefore, Alice might consider it safer to never sign. This kind of reasoning can be captured by using Nash equilibrium as*

the solution concept. Only $(Stop, Stop)$ *is a Nash equilibrium, i.e.,* $NE(G_2) = \{(Stop, Stop)\}$*. When the objective is* $\gamma = \{(Sign, Sign)\}$*, then* $NE(G_2) \subseteq \gamma$ *does not hold, i.e.,* G_2 *is not correct with respect to* γ *under Nash equilibrium as solution concept.*

However, if we think that Alice may be willing to take risks in order to obtain a better outcome, using e.g. Halpern's maximal perfect collaborative equilibrium [HR10] as the solution concept is more appropriate. As we have that $MPCE(G_2) = \{(Sign, Sign)\} \subseteq \gamma$*, the protocol is correct with respect to* γ *under* $MPCE$*.*

Finally, consider the protocol game G_1 *in Figure 6.7a. In* G_1*, we have* $NE(G_1) = Undom(G_1) = MPCE(G_1) = \{(Sign, Sign)\}$*. This implies that when we assume* $\gamma = \{(Sign, Sign)\}$*, the protocol is correct under Nash equilibrium, equilibrium in undominated strategies, and maximal perfect cooperative equilibrium.*

Now consider the game G_3 *with objective* $\gamma = \{(Sign_1, Sign_1), (Sign_2, Sign_2)\}$*. As we have Nash equilibria* $NE(G_3) = \{(Sign_1, Sign_1), (Sign_2, Sign_2)\}$*, both Nash equilibria satisfy the objective, i.e.,* $NE(G_3) \subseteq \gamma$*, so the protocol is correct with respect to Nash equilibrium. However, Nash equilibrium assumes that the agents have a way to coordinate on their strategy (for example by considering how the agents played in previous games). If no such coordination mechanism exists, other solution concepts, such as equilibrium in undominated strategies, might be more appropriate. As* $Undom(G_3) = \{Sign_1, Sign_2\} \times \{Sign_1, Sign_2\}$*, we have that* $NE(G_3) \not\subseteq \gamma$*, so the protocol is incorrect with respect to the solution concept.*

The above example highlights that, for different situations, different solution concepts are appropriate.

The solution concept models both the rationality assumptions for the involved agents and the abilities to coordinate strategies, etc.

6.3.2 Unknown Incentives

In the previous section, we studied correctness of a protocol when a utility profile is given. However, often the exact utility profiles are unknown. One way out is to require the protocol to be correct for *all possible* utility profiles.

Definition 6.8. *A protocol is* valid *with respect to objective* γ *under solution concept* SC*, written* $P \models_{SC} \gamma$*, if and only if* $(P, u) \models_{SC} \gamma$ *holds for all strategy profiles* u*.*

Protocol validity *is the following decision problem:*

- ***Input:*** *A protocol* P*, an objective* γ *and a solution concept* SC*.*
- ***Question:*** *Does* $P \models_{SC} \gamma$ *hold?*

Now we show that, under some reasonable assumptions, no protocol is valid. The first assumption is that not all strategy profiles of the protocol are correct, i.e., $\gamma \neq \Sigma$. The second assumption is that the solution concept is *closed under permutation of utilities*, which can be defined as follows:

Definition 6.9. *Let $G = (N, \Sigma, (u_1, \ldots, u_n))$. Let $\pi = (\pi_1, \ldots, \pi_n)$, where for all $i \in N$, $\pi_i : \Sigma_i \to \Sigma_i$ is a permutation on Σ_i. For conciseness of notation, sometimes we write $\pi((s_1, \ldots, s_n))$ for $(\pi_1(s_1), \ldots, \pi_n(s_n))$. A solution concept is closed under permutation of utilities if $s \in SC((N, \Sigma, (u'_1, \ldots, u'_n)))$ if and only if $\pi(s) \in SC((N, \Sigma, (\pi(u'_1), \ldots, \pi(u'_n))))$.*

All solution concepts that we are aware of are closed under permutation of utilities.

Theorem 6.1. *If $\gamma \neq \Sigma$ and SC is closed under permutation of utilities, then $P \not\models_{SC} \gamma$.*

Proof. When a solution concept is closed under permutation of utilities, and all outcomes have the same utility, then either all strategy profiles are rational, or none of them.

Let SC be a solution concept, and let $G = (N, \Sigma, (u_1, \ldots, u_n))$ be a normal-form game. If there exists $r_1, \ldots, r_n \in \mathbb{R}$ such that $u_i(s) = r_i$ for all $i \in N$ and $s \in \Sigma$, then either $s \in SC(G)$ for all s, or $s \notin SC(G)$ for all s.

Let $G = (N, \Sigma, (u_1, \ldots, u_n))$ be a game, let $s, s' \in \Sigma$, and let SC be closed under permutation of utilities. Let $\pi = (\pi_i, \ldots, \pi_n)$, where each π_i is a permutation on Σ_i such that $\pi(s) = s'$. As $u_i(s) = r_i$ for all $i \in N$, $\pi(G) = G$. As $s \in SC(G)$ if and only if $\pi(s) \in SC(\pi(G))$, by $\pi(G) = G$ and $\pi_i(s) = s'$, we obtain $s \in SC(G)$ if and only if $s' \in SC(G)$. Therefore, either $s \in u(G)$ for all s, or $s \notin u(G)$ for all s.

Let G be a game, SC be a solution concept closed under permutation of utilities, and γ be an objective. Fix u such that $u_i(s) = (0, 0)$ for all $i \in N$ and $s \in \Sigma$. Either $s \in SC(G)$ for all $s \in \Sigma$, or $s \notin SC$ for all $s \in \Sigma$. Assume $s \in SC(G)$ for all $s \in \Sigma$. As $\gamma \neq \Sigma$, there exists $s' \in \Sigma \backslash \gamma$. Then $s' \in SC(G)$ and not $s' \in \gamma$, so $SC(G) \not\subseteq \gamma$, and therefore $P \not\models_{SC} \gamma$. Now assume $s \notin SC(G)$ for all $s \in \Sigma$. Then $SC(G) = \emptyset$, so $P \not\models_{SC} \gamma$. □

Thus, correctness for all distributions of incentives is equivalent to correctness in all possible runs—as we argued before, requiring this can be too strong. In the next section we look at the case where a subset of agents D, called the *defenders* of the protocol, have a genuine interest in achieving the objective of the protocol.

An interesting special case of Theorem 6.1 appears in a study of rational secret sharing: Asharov and Lindell [AL11] proved that the *length* (number of rounds) of a protocol for rational secret sharing must depend on the utilities of the involved agents for the possible protocol outcomes. In particular, there can be no single protocol which works for every possible set of incentives of the agents. Their result even holds under some plausibility assumptions on the agents' incentives (i.e., agents prefer to learn the secret over not learning it, etc.).

6.3.3 Defendability of Protocols

In many circumstances, requiring that the protocol is correct with respect to *all* preferences is too pessimistic: In a way, this is equivalent to assuming that all the participants are playing *against* the security objective. Typical analysis of

a protocol implicitly assumes some participants to be aligned with its purpose. E.g., one usually assumes that communicating parties are usually interested in exchanging a secret without the eavesdropper getting hold of it, that a bank wants to prevent web banking fraud etc. In this chapter, we formalize this idea by assuming a subset of agents, called the *defenders* of the protocol, to be in favor of its objective. Our new definition of correctness says that a protocol is correct if and only if it is correct with respect to every utility profile in which the preferences of all defenders comply with the objective.

Definition 6.10 (Supporting an objective)**.** *A group of agents $D \subseteq N$* supports *the objective γ in game (N, Σ, u) if for all $i \in D$, if $s \in \Sigma \cup \gamma$ and $s' \in \Sigma \backslash \gamma$ then $u_i(s) > u_i(s')$.*

A protocol represented as game frame $P = (N, \Sigma, \Omega, o)$ is correct with defenders *D, written $P \models_{SC} [D]\gamma$, if $(P, u) \models_{SC} \gamma$ for all utility profiles u such that D support γ in game $\Gamma((N, \Sigma, \Omega, o), u)$.*

Guarded protocol validity *is the following decision problem:*

- ***Input:*** *Protocol P, objective γ, set of agents D, and solution concept SC.*
- ***Question:*** *Does $(P, u) \models_{SC} [D]\gamma$ hold?*

For effectivity, it makes sense to require that both Alice (Bob) is a defender of the goal that she (he) obtains a contract, as we cannot expect that the contract will get signed if any of the agents prefer not to sign. In the case of fairness for Alice, we can expect Alice to be a defender of the property. Bob, on the other hand, cannot be expected to defend this. The situation for fairness for Bob is symmetric.

We proceed by investigating the borderline cases, where either none or all of the agents are defenders. The following theorem shows that if none of the agents are defenders, the definition is equivalent to ordinary protocol validity.

Proposition 6.1. *If P is a game frame and SC is a solution concept, we have that $P \models_{SC} [\emptyset]\gamma$ if and only if $P \models_{SC} \gamma$.*

Proof. This follows directly from the definitions of validity and validity with defenders. □

Proof. Let $P = (N, \Sigma, \Omega, o)$ be a game frame and SC be a solution concept. Assume $P \models_{SC} []\gamma$. This is equivalent to $(P, u) \models_{SC} \gamma$ for all utility profiles u such that $\emptyset$ support γ in game $\Gamma((N, \Sigma, \Omega, o), u)$. This is in turn equivalent to $(P, u) \models_{SC} \gamma$ for all utility profiles u such that for all $i \in \emptyset$, if $s \in \Sigma \cup \gamma$ and $s' \in \Sigma \backslash \gamma$ then $u_i(s) > u_i(s')$ in game $\Gamma((N, \Sigma, \Omega, o), u)$. This is in turn equivalent to $(P, u) \models_{SC} \gamma$ for all utility profiles u. This is in turn equivalent to $P \models_{SC} \gamma$. □

If all agents are defenders, any protocol is correct, as long as there exists a rational strategy profile, and the solution concept does not select strategy profiles that are *strongly Pareto-dominated*, i.e., strategy profiles for which there exists another strategy profile that is preferred by all agents.

We formalize this in Theorem 6.2.

Definition 6.11. *A strategy profile $s \in \Sigma$ is Pareto-dominated if there exists $s' \in \Sigma$ such that $u_i(s') > u_i(s)$ for all $i \in N$.*

Definition 6.12. *A solution concept is* weakly Pareto *if and only if it never selects a strongly Pareto dominated outcome (that is, one such that there exists another outcome strictly preferred by all the agents). It is* efficient *if and only if it never returns the empty set.*

Theorem 6.2. *If P is a game frame and SC an efficient weakly Pareto solution concept then $P \models_{SC} [N]\gamma$.*

Proof. Let $P = (N, \Sigma, \Omega, o)$ be a game frame, and let u be a utility function such that D support γ in game (P, u). We have $SC(P, u) \neq \emptyset$ by assumption. Now we prove that $SC(P, u) \subseteq \gamma$. Assume $s \in SC(P, u)$ and $s \notin \gamma$. Let $s' \in \gamma$. Then $u_i(s') > u_i(s)$ for all $i \in N$. However, this implies that $s \notin SC(P, u)$, which is a contradiction. □

A note on existing solution concepts: Nash equilibrium is neither weakly Pareto nor efficient (the Hi/Lo from game Figure 6.8a being a counterexample). Equilibrium in dominant strategies is weakly Pareto but not necessarily efficient. Stackelberg equilibrium and Halpern's maximum-perfect cooperative equilibrium are weakly Pareto and efficient. Also, backward induction and, more generally, subgame-perfect equilibrium in perfect information games are weakly Pareto and efficient.

Given a protocol, a solution concept and an objective, it is interesting to determine the smallest set of defenders with which the protocol is correct. We show that correctness of a protocol is monotonic with respect to the set of defenders.

Theorem 6.3. *For every $d \in N$ and set of defenders $D \subseteq N$, if $P \models_{SC} [D]\gamma$ then $P \models_{SC} [D \cup \{d\}]\gamma$.*

Proof. Let $P = (N, \Sigma, \Omega, o)$ be a protocol represented as game frame. Assume that $P \models_{SC} [D]\gamma$. Then we have $(P, u) \models_{SC} [D]\gamma$ for all preference profiles u in which D defend γ. This means that for all $i \in D$, if $s \in \gamma$ and $s' \in \Sigma \backslash \gamma$ then $u_i(s) > u_i(s')$. Therefore, also for all $i \in D \cup \{d\}$, we have that if $s \in \gamma$ and $s' \in \Sigma \backslash \gamma$ then $u_i(s) > u_i(s')$. This implies that $(P, u) \models_{SC} [D \cup \{i\}]\gamma$ for all preference profiles u in which $D \cup \{i\}$ defend γ. Therefore, we have $P \models_{SC} [D]\gamma$. □

This justifies the following definition.

Definition 6.13 (Game-theoretic security level)**.** *The* game-theoretic security level *of protocol P as the antichain of minimal sets of defenders that make the protocol correct.* Defender determination *is the following function problem:*

- ***Input:*** *A protocol P, an objective γ, and a solution concept SC.*
- ***Question:*** *Compute a set D such that $(P, u) \models_{SC} [D]\gamma$, and there does not exist $D' \subseteq D$ such that $(P, u) \models_{SC} [D']\gamma$.*

	t_1	t_2
s_1	hi, hi	$0, 0$
s_2	$0, 0$	lo, lo

(a)

	t_1	t_2	t_3
s_1	hi, lo	lo, hi	$0, 0$
s_2	lo, hi	hi, lo	$0, 0$
s_3	$0, 0$	$0, 0$	$0, 0$

(b)

Figure 6.8: (a) HiLo game for 2 agents; (b) Extended matching pennies. In both games, we assume that $hi > lo > 0$, e.g., $hi = 100$ and $lo = 1$

6.4 Characterizations of Defendability

Not much can be said about defendability under arbitrary solution concepts. In this section, we turn to properties that can be defended if agents' rationality is defined by Nash equilibrium. In what follows, we will call objective γ *nontrivial* if and only if $\emptyset \neq \gamma \neq \Sigma$. First, we observe that no protocol is valid under Nash equilibrium (NE) for any nontrivial objective.

Theorem 6.4. *$P \not\models_{NE} \gamma$ for any nontrivial γ.*

Proof. Let $s \notin \gamma$ (arbitrary). We construct utility profile u as follows: $u_i(s) = 1$ for all $i \in N$; $u_i(s') = 0$ for all $i \in N, s' \neq s$. Clearly, $NE(P, u) = \{s\}$, so $NE(P, u)$ is nonempty but not included in γ. □

Do things get better if we assume some agents to be in favor of the security objective? In the rest of this section, we study the extreme variant of the question, i.e. defendability by the grand coalition N. Note that, by Theorem 6.3, nondefendability by N implies that the objective is not defendable by any coalition at all.

Our first result in this respect is also negative: we show that in every game frame there are nontrivial objectives that are not defendable under Nash equilibrium.

Theorem 6.5. *Let P be a game frame with at least two agents and at least two strategies per agent. Moreover, let γ be a singleton objective, i.e., $\gamma = \{\omega\}$ for some $\omega \in \Sigma$. Then, $P \not\models_{NE} [N]\gamma$.*

Proof. Assume without lose of generality that $N = 2$, $\Sigma_1 = \{s_1, s_2\}$, $\Sigma_2 = \{t_1, t_2\}$, and $\gamma = \{(s_1, t_1)\}$. Now, consider the utility function u^{hl} of the well known HiLo game (Figure 6.8a). Clearly, u^{hl} is consistent with γ. Moreover, $SC(P, u^{hl}) \neq \emptyset$. On the other hand, $SC(P, u^{hl}) = \{(s_1, t_1), (s_2, t_2)\} \not\subseteq \gamma$, which concludes the proof. □

To present the general result that characterizes defendability of security objectives under Nash equilibrium, we need to introduce additional concepts. In what follows, we use $s[t_i/i]$ to denote $(s_1, \ldots, s_{i-1}, t_i, s_{i+1}, \ldots, s_N)$.

Definition 6.14 (Deviation closure)**.** *Let γ be a set of outcomes (strategy profiles) in P. The* deviation closure *of γ is defined as $Cl(\gamma) = \{s \in \Sigma \mid \exists i \in N, s'_i \in \Sigma_i \,.\, s[s'_i/i] \in \gamma\}$.*

$Cl(\gamma)$ extends γ with the strategy profiles that are reachable by unilateral deviations from γ (and vice versa). Thus, $Cl(\gamma)$ can be seen as the set of outcomes that are "connected" to γ in the sense of Nash equilibrium. The following notion captures strategy profiles that can be used to construct sequences of unilateral deviations ending up in a cycle.

Definition 6.15 (Strategic knot)**.** *A* strategic knot *in γ is a subset of strategy profiles $S \subseteq \gamma$ such that there is a permutation $(s^1, \ldots, s^k)$ of S where each $s^{j+1} = s^j[s_i^{j+1}/i]$ for some $i \in N$, and $s^k = s^j[s_i^k/i]$ for some $i \in N, j < k$.*

We can now state the main result in this section.

Theorem 6.6. *Let P be a finite game frame with $N \geq 2, |\Sigma_1| \geq 3, |\Sigma_2| \geq 3$, and γ a nontrivial objective in P. Then, $P \models_{NE} [N]\gamma$ if and only if $Cl(\gamma) = \Sigma$ and there is a strategy profile in γ that belongs to no strategic knots in γ.*

Proof. "⇒" Let $P \models_{NE} [N]\gamma$, and suppose that $Cl(\gamma) \neq \Sigma$. Thus, there exists $\omega_0 \in \Sigma$ which is not in $Cl(\gamma)$. Consider a HiLo-style utility function $u(s) = hi$ if $s \in \gamma$, lo if $s = \omega_0$, and 0 otherwise (for some values $hi > lo > 0$). Clearly, ω_0 is a Nash equilibrium in (P, u), and thus $NE(P, u) \neq \emptyset$) but also $NE(P, u) \not\subseteq \gamma$), which is a contradiction.

Suppose now that $Cl(\gamma) = \Sigma$ but every $s \in \gamma$ belongs to a strategic knot. We construct the utility function akin to the extended matching pennies game (Figure 6.8b), i.e., for every node s in a ring $u_i(s) = hi$ for the agent i who has just deviated, and lo for the other agents (if a node lies on several knots, we need to assign several different hi values in a careful way). Moreover, $u_i(s) = 0$ for all $i \in N, s \notin \gamma$. Clearly, u_i is consistent with γ for every $i \in N$. On the other hand, no $s \in \Sigma$ is a Nash equilibrium: if s is outside of γ then there is a profitable unilateral deviation into γ, and every s inside γ lies on an infinite path of rational unilateral deviations. Thus, $NE(P, u) = \emptyset$. Since γ is nontrivial, we have $P \not\models_{NE} [N]\gamma$, a contradiction again.

⇐ Assume that $Cl(\gamma) = \Sigma$ and $s \in \gamma$ is a strategy profile that belongs to no strategic knot in γ. Let u be a utility function such that for every $i \in N, s \in \gamma, s' \in \Sigma \setminus \gamma$ it holds that $u_i(s) > u_i(s')$. Take any $\omega \notin \gamma$. Since $\omega \in Cl(\gamma)$, there is an agent i with a unilateral deviation to some $s \in \gamma$. Note that $u_i(\omega) < u_i(s)$, so $\omega \notin NE(P, u)$. Thus, $NE(P, u) \subseteq \gamma$. Moreover, s is a Nash equilibrium or there is a sequence of unilateral deviations leading from s to a Nash equilibrium (since P is finite and s does not lie on a knot). Thus, also $NE(P, u) \neq \emptyset$, which concludes the proof. □

6.5 Examples

6.5.1 The ASW Protocol

The contract-signing protocol P_{ASW}, introduced in [ASW98], uses co-called *commitments*, which are legally binding "declarations of intent" by Alice and Bob to sign the contract. The protocol is fair and *optimistic*, i.e., it involves the TTP only

if a problem occurs. The protocol operates as follows: 1) Alice sends a *commitment* cm_A to Bob; 2) Bob sends his commitment cm_B to Alice; 3) Alice sends the contract sc_A, digitally signed with her signature, to Bob; 4) Bob sends the contract sc_B, signed with his signature, to Alice. If one of these messages is not sent by the corresponding signer, the other party may contact the TTP:

- If Alice does not receive a commitment from Bob, she can contact the TTP with an *abort request*, which instructs the TTP to mark this session of the protocol as aborted;
- If Bob does not receive Alice's signature, but has her commitment, he can send a *resolve request* to the TTP, who then issues a *replacement contract* (a document that is legally equivalent to the contract signed by Alice), unless Alice has sent an abort request earlier,
- If Alice does not receive Bob's signature, but has his commitment, she can send a *resolve request* to the TTP as well, which allows her to receive a replacement contract.

It can be shown that the protocol is fair. It is also *balanced*, if neither Alice nor Bob can drop or delay messages from the other signer to the TTP.

We denote with u the usual utilities for the involved agents, i.e., $u_A(\{\varphi_B\}) = 0$, $u_A(\emptyset) = 1$, $u_A(\{\varphi_A, \varphi_B\}) = 2$, $u_A(\{\varphi\}) = 3$, and symmetrically for Bob. Their utilities reflect that each signer wants to avoid giving his signature without receiving the one of the other signer, but prefers a fair exchange of signatures over no exchange result, but would prefer to only obtain the other signer's signature without having to sign himself. We assume that the TTP is reliable, i.e., will not stop the protocol run and thus has only a single strategy. Applying the definitions in Section 6.3.2, one can easily show the following: If SC is either Nash equilibrium or Undominated Strategies, we have

1. $P_{\text{ASW}} \models_{SC} [\{\text{Bob}\}]\{\emptyset, \{\varphi_B\}, \{\varphi_A, \varphi_B\}\}$,
2. $P_{\text{ASW}} \models_{SC} [\{\text{Alice}\}]\{\emptyset, \{\varphi_A\}, \{\varphi_A, \varphi_B\}\}$.

It is clear that if we allow the TTP to stop the protocol at any time, then the protocol does not guarantee fairness anymore. However, if Bob wants the protocol to be fair, then he can ensure fairness by simply sending a signed contract to Alice as soon as he receives her signature. Clearly, Alice alone (without an honest TTP to assist her) cannot achieve fairness. Hence the game-theoretic security level of the ASW protocol is the set $\{\{\text{Bob}\}, \{\text{TTP}\}\}$. This holds for both Nash equilibrium and undominated strategies.

6.5.2 A Protocol with a Non-deterministic TTP

We consider the contract signing protocol P_{Aiz} suggested by Aizatulin in [Aiz08]. This protocol is an extension of the well-known contract signing protocols ASW and GJM. It is well-known [CKS01] that the latter two protocols are unbalanced

against an attacker who can drop or delay messages on the network, since then the attacker can always enforce that his request is the first one to reach the trusted third party.

The protocol P_{Aiz} counteracts this problem with two measures:

- To the protocol structure of ASW and GJM, it adds an additional round of *pre-commitments*. These are handled in a similar way as commitments in the normal ASW/GJM protocol flow. A protocol exchange now consists of

  1. An exchange of *pre-commitments*,
  2. an exchange of *commitments*,
  3. the exchange of the actual contracts.

  The pre-commitments play a similar role as commitments; if a party did send his or her commitment, but does not receive a commitment from the other party, he can try to resolve the contract signing with help of the TTP.

- The trusted third party is *non-deterministic*: When a resolve request is made with a pre-commitment, the TTP may either accept or decline the request (if, however, a resolve request is performed with an actual commitment, the TTP always issues a valid contract).

The rules of the TTP additionally handle exceptions as dishonest behavior of principals that, for example, send an abort request to the TTP, but still continue the regular protocol run. The protocol is fair due to a similar reasoning as the ASW protocol above (we assume that the TTP does not "misbehave", i.e., the TTP always follows the protocol and does not stop working). Since the TTP is non-deterministic, it has a choice, i.e., it may follow a strategy. Clearly, the TTP can side with an attacker and just accept/request on his or her behalf. However, if the TTP supports balance, the protocol is indeed balanced, as we now show.

Let $\gamma_{\mathsf{bal\text{-}Alice}}$ denote the runs of the protocol that are balanced for Alice, i.e., where no situation appears in which Bob has both a strategy to ensure that he gets a contract, and a (different) strategy in which Alice does not obtain one. Similarly, let $\gamma_{\mathsf{bal\text{-}Bob}}$ denote the runs that are balanced for Bob.

1. $P_{Aiz} \models_{SC} [\{\mathrm{TTP}\}]\gamma_{\mathsf{bal\text{-}Bob}}$,
2. $P_{Aiz} \models_{SC} [\{\mathrm{TTP}\}]\gamma_{\mathsf{bal\text{-}Alice}}$,

The first point follows since the TTP can always ignore Alice's requests and act in Bob's favor. Analogously, the second point is true since the TTP can rule according to Alice's wishes instead. To achieve balance for both Alice and Bob, a natural strategy for the TTP is to randomize its replies to requests involving a pre-commitment: Assume that the TTP accepts or rejects each such request with probability $\frac{1}{2}$. Then, clearly, neither Alice nor Bob can have both a strategy to obtain a contract themselves as well as one to stop the other from obtaining a contract: One of these strategies must rely on the TTP ruling in their favor, which only happens with probability $\frac{1}{2}$. In [ASW09], a probabilistic version of the protocol was analyzed and shown to guarantee probabilistic balance in a computational setting.

6.6 Related Work

There are several meeting points of security protocols and game theory. Some researchers have considered protocol execution as a game with the very pessimistic assumption that the only goal of the other participants ("adversaries") is to break the intended security property of the protocol. In the pessimistic analysis of correctness, a protocol is deemed correct if the "honest" participants have a strategy such that, for all strategies of the other agents, the objective of the protocol is satisfied (cf. e.g. [KR02]). Recently, protocols have been analyzed with respect to game-theoretic notions of rationality [Mic10, ACH11] where preferences of participants are taken into account. An overview of connections between cryptography and game theory is given in [Kat08], focusing mainly on correlated equilibria in multi-party computation protocols. Another overview [AMNO07] presents arguments suggesting that study of incentives in security applications is crucial.

Game-theoretic concepts have been applied to analysis of specific security properties in a number of papers. Kremer and Raskin [KR03] used graph games to formally verify non-repudiation protocols. However, their method did not use any model of incentives nor rationality. Buttyán, Hubaux and Čapkun [BHC04] model games in a way similar to as is done in this chapter, and also use incentives to model the behavior of agents. However, they restrict their analysis to strongly Pareto-optimal Nash equilibria which is not necessarily a good solution concept for security protocols. First, it is unclear why agents would *individually* converge to a strongly Pareto-optimal play. Moreover, in many protocols it is unclear why agents would play a Nash equilibrium in the first place. Our method is more general, as we use the solution concept as a parameter to our analysis.

Asharov et al. [ACH11] use game theory to study gradual-release fair exchange protocols, i.e., protocols in which at any round, the probability of any party to predict the item of the other player increases only by a negligible amount. They model this in a game-theoretical setting, where in every round, the player can either continue or abort. In every round, the item of the other player is predicted. The situation where the player predicts correctly and the other one does not has the highest utility, and the situation where the player predicts incorrectly and the other one predicts correctly the lowest. Then a protocol is said to be game-theoretically fair if the strategy that never aborts the protocol is a computational Nash-equilibrium. It is shown that no protocol exists that is both fair and effective, but that if effectiveness is not required, game-theoretic fairness is in fact achievable. They also show that their analysis allows for solutions that are not admitted by the traditional cryptographic definition.

Groce and Katz [GK12] show that if agents have a strict incentive to achieve fair exchange, then gradual-release fair exchange without TTP is in fact possible under the assumption that the other agents play rational. Fairness without TTP is impossible for dishonest partners. However, it might be possible for rational partners. Chadha et. al [CMSS05] show that in any fair, optimistic, timely protocol, there exists a point where one player has a strategy to determine whether or not to complete the protocol and obtain a contract. The strongest property attainable is provable advantage, i.e. abuse-freeness. Although they reason about strategies, they do not model incentives explicitly, and also do not take different solution

concepts into account. Syverson [Syv98] presents a *rational exchange* protocol for which he shows that "enlightened, self-interested parties" have no reason to cheat.

Chatterjee & Raman [CR10] use assume-guarantee synthesis for synthesis of contract signing protocols. Our analysis is strictly more general, as we can model assume-guarantee synthesis by assuming that A is a defender of φ_A, while B is a defender of φ_B, and both A and B are defenders of the joint goal φ_C. Finally, in [FS10], a logic for modeling coordination abilities between agents is presented, but incentives are not taken into account. Ghaderi, Levesque & Lespérance [GLL07] also study coordination and applies iterated elimination of dominated strategies.

In summary, rationality-based correctness of protocols has been studied in a number of papers, but usually with a particular notion of rationality in mind. In contrast, we define a concept of correctness where a game-theoretic solution concept is a parameter of the problem. Even more importantly, our concept of *defendability* of a security property is completely novel. The same applies to our characterizations of defendable properties under Nash equilibrium.

6.7 Conclusion

We have proposed a methodological framework for analyzing security protocols and other interaction protocols that takes into account the incentives of agents. This framework allows for a more fine-grained analysis of such protocols. We have obtained characterization results for defendability under Nash equilibria. Our analysis allows for an easy combination of different requirements, as we saw in Chapter 3. For example, to verify that a protocol is correct with respect to both fairness and effectivity, it suffices to check both fairness and effectivity. Not all approaches allow for an easy combination of different requirements [JMM12].

In the future, we plan to select solution concepts that can be expressed in the logic ATL [AHK02], and use tools to formally verify protocols with our methodology. Furthermore, we want to focus on the coordination problem, and study which assumptions on rationality, i.e., which solution concepts, are most reasonable in practice. Finally, we are going to extend our analysis to models of games with imperfect information.

7

Farsighted Pre-equilibria

Abstract. Nash equilibrium is based on the idea that a strategy profile is stable if no player can benefit from a unilateral deviation. We observe that some locally rational deviations in a strategic-form game may not be profitable anymore if one takes into account the possibility of further deviations by the other players. As a solution, we propose the concept of farsighted pre-equilibrium, which takes into account only deviations that do not lead to a decrease of the player's utility even if some other deviations follow. While Nash equilibria are taken to include plays that are certainly rational, our pre-equilibrium is supposed to rule out plays that are *certainly irrational*. We prove that positional strategies are sufficient to define the concept, study its computational complexity, and show that pre-equilibria correspond to subgame-perfect Nash equilibria in a meta-game obtained by using the original payoff matrix as arena and the deviations as moves.

7.1 Introduction

The optimal strategy for an agent depends on his prediction of the other agents' behavior. For example, in security analysis, some predictions of the users' behavior, or even of the behavior of the attacker, can be useful when designing a particular solution. However, if the users or attackers do not behave in the predicted way, this solution might give rise to new vulnerabilities. We therefore weaken the assumptions made by agents when playing Nash equilibrium, and introduce a new solution concept based on these weaker assumptions.

Nash equilibrium (e.g. [OR94]) says that a play is stable when, if the agents knew what the others are going to do, they would not deviate from their choices unilaterally. Conversely, if some agent can beneficially deviate from strategy profile s, then the profile is assumed to describe irrational play. In this chapter, we point out that some of these deviations may not be profitable anymore if one takes into account the possibility of further deviations from the opponents. As a solution, we propose the concept of *farsighted pre-equilibrium* (*FPE*), which takes into account only those deviations of agent i that do not lead to decrease of i's utility, even if some other deviations follow. In consequence, we argue that the notion of irrational play can be meaningfully relaxed.

We call the new concept *pre-equilibrium*, because we do not imply that all FPEs are necessarily stable. Our point is rather that all strategy profiles outside FPEs are certainly *un*stable: a rational agent should deviate even if he considers it possible that other agents react to his change of strategy. Formally, FPE is strictly weaker than Nash equilibrium, with the following intuition: Nash equilibria correspond

to play which is certainly rational, strategy profiles that are *not* pre-equilibria are certainly irrational, and the profiles in between can be rational or not, depending on the circumstances.

The term "farsighted" refers to the type of reasoning about strategic choice that agents are supposed to conduct according to FPE. Unlike Nash equilibrium, which assumes "myopic" reasoning (only the immediate consequences of a deviation are taken into account), farsighted pre-equilibrium looks at further consequences of assuming a particular choice. This type of strategic reasoning has been already studied for coalitional games in [NM44, Har74, Chw94]. There have been also some attempts at farsighted stability in noncooperative games [SM05, Nak07], but, as we argue in Section 7.5, they were based on intuitions from coalitional game theory, and are incompatible with the setting of noncooperative games.

Our assumptions about the way in which agents react to another agent's deviation are minimal: we only assume that the reactions are locally rational. Our view of local rationality is standard for noncooperative games, i.e., it concerns an *individual* change of play that increases the payoff of the deviating agent. In particular, we do not take into account scenarios where a coalition of agents makes a sequence of changes that leads to a beneficial state, but leads through nodes where the payoff of some members of the coalition decreases. As we see it, such a scenario can be rational only when the coalition can commit to executing the sequence, which is not possible in noncooperative games.

An interpretation of Nash equilibrium is that an agent forms an expectation about the other agents' behavior based on his past experience of playing the game [OR94]. Then he chooses his best response strategy to maximize his immediate gain in the next instance of the game, assuming that this move will not influence future plays of the game. In other words, it is assumed that the other agents do not best respond to a deviation from the expectation when the game is repeated. In contrast, in repeated games [MS06], it is assumed that once an agent decides to deviate, the deviation will be observed by the opponents, and they will adapt to it accordingly. Then, the agent would observe and adapt to their change of behavior, and so on.

Farsighted pre-equilibrium is very similar to the standard setting of infinitely repeated games (with a sufficiently small discount factor). In both settings, it is assumed that the agent considers how his behavior can affect the behavior of the other agents in subsequent games. However, while in repeated games it is assumed that opponents *always* best respond, in FPE no assumption is made about the opponents at all in this respect. We are looking for a *weak* notion of rationality, and restricting the opponents' possible responses would make the solution concept stronger. Therefore, our deviations need to succeed against agents that do not best respond at all (as assumed by Nash equilibrium), against agents that best respond locally, against agents that best respond farsightedly (as in the standard setting of repeated games), and against combinations of such agents.

We do not pursue the perspective of repeated games further in this chapter, but rather leave it for future work.

This chapter is organized as follows. We begin by defining the concept of farsighted pre-equilibrium formally and discussing some examples in Section 7.2. We investigate how the concept behaves on the benchmark case of the n-person Pris-

oner's Dilemma, provide an alternative characterization of FPE, and propose a polynomial algorithm for verifying pre-equilibria. In Section 7.3, we given an alternative characterization of FPE. In Section 7.4, we show that FPEs can be seen as subgame-perfect solutions of specific extensive-form games ("deviation games"). Finally, we compare our solution concept to existing work (Section 7.5), and conclude in Section 7.6.

7.2 Farsighted Pre-Equilibria

We begin by presenting the central notions of our solution concept.

7.2.1 Deviation Strategies and Farsighted Stability

Let $G = (N, \Sigma_1, \ldots, \Sigma_n, out_1, \ldots, out_n)$ be a strategic game with $N = \{1, \ldots, n\}$ being a set of agents, Σ_i a set of strategies of agent i, and $out_i : \Sigma \to \mathbb{R}$ the payoff function for agent i where $\Sigma = \Sigma_1 \times \cdots \times \Sigma_n$ is the set of strategy profiles. We use the following notation: s_i is agent i's part of strategy profile s, s_{-i} is the part of $N \setminus \{i\}$, and $s \to_i s'$ denotes agent i's deviation from strategy profile s to s' (with the obvious constraint that $s'_{-i} = s_{-i}$). Sometimes, we write $(out_1(s), \ldots, out_n(s))$ instead of s.

Definition 7.1 (Locally rational, F_i-compatible)**.** *Deviation $s \to_i s'$ is* locally rational *if and only if $out_i(s') > out_i(s)$. Function $F_i : \Sigma^+ \to \Sigma$ is a* deviation strategy for agent i *if and only if for every finite sequence of profiles $s^1, \ldots, s^k$ we have that $s^k \to_i F_i(s^1, \ldots, s^k)$ is locally rational or $F_i(s^1, \ldots, s^k) = s^k$. A sequence of locally rational deviations $s^1 \to \ldots \to s^k$ is F_i-*compatible *if and only if $s^n \to_i s^{n+1}$ implies $F_i(s^n) = s^{n+1}$ for every $1 \leq n < k$.*

Locally rational deviations turn G into a graph in which the transition relation corresponds to Nash dominance in G. Deviation strategies specify how an agent can (rationally) react to rational deviations done by other agents.

Definition 7.2 (Farsighted pre-equilibrium)**.** *Strategy profile s is a* farsighted pre-equilibrium *(*FPE*) if and only if there is no agent i with a deviation strategy F_i such that: 1) $out_i(F_i(s)) > out_i(s)$, and 2) for every finite F_i-compatible sequence of locally rational deviations $F_i(s) = s^1 \to \ldots \to s^k$ we have $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$.*

This means that a strategy profile s is potentially *un*stable if there is a deviation strategy of some agent i such that the first deviation is strictly advantageous, and no matter how the other agents react to his deviations so far, i can always recover to a profile where he is not worse off than he was originally in s. Moreover, s is still unstable if the other agents reply to this recovery as long as i has a new recovery, and so on.

Example 7.1. *Consider the following games:*

(A)	C	D
C	$(\mathbf{7}, \mathbf{7})$	$(0, 8)$
D	$(8, 0)$	$(\mathbf{1}, \mathbf{1})$

(B)	L	C	R
T	$(1, 1)$	$(\mathbf{3}, \mathbf{3})$	$(0, 0)$
B	$(0, 0)$	$(4, 0)$	$(\mathbf{2}, \mathbf{2})$

The farsighted pre-equilibria are printed in bold font. Let us first look at game (A), which is also known as the Prisoner's Dilemma. *The locally rational deviations are* $(7,7) \to_1 (8,0)$, $(0,8) \to_1 (1,1)$, $(7,7) \to_2 (0,8)$ *and* $(8,0) \to_2 (1,1)$. *This implies that* $(1,1)$ *is an FPE because there is no agent* i *with a deviation strategy* F_i *such that* $out_i(F_i(1,1)) > out_i(1,1)$. *On the other hand,* $(8,0)$ *is not an FPE because* $F_2(\ldots,(8,0)) = (1,1)$ *is a valid deviation strategy. By symmetry,* $(0,8)$ *is neither an FPE. Finally we show that* $s = (7,7)$ *is an FPE. All deviation strategies* F_1 *for agent 1 with* $out_1(F_1(7,7)) > out_1(7,7)$ *specify* $F_1(7,7) = (8,0)$. *Still, agent 1 cannot recover from the* F_1*-compatible sequence of locally rational deviations* $(7,7) \to_1 (8,0) \to_2 (1,1)$ *which makes his payoff drop down to 1. The same holds for deviation strategies of agent 2 by symmetry. Therefore,* $(7,7)$ *is an FPE.*

In game B, $(3,3)$ *and* $(2,2)$ *are FPEs for similar reasons. Note than* $(1,1)$ *is not an FPE, as agent 1 can deviate* $(1,1) \to_1 (3,3)$, *and recover from the follow-up* $(3,3) \to_1 (4,0)$ *by playing* $(4,0) \to_1 (2,2)$, *resulting in a final utility 2, which is still higher than the initial utility 1.*

The following theorem shows that FPE is strictly weaker than Nash equilibrium.

Theorem 7.1. *Every Nash equilibrium is an FPE.*

Proof. Assume that s is a Nash equilibrium. Then there exists no deviation $s \to_i s'$ to a strategy profile s' such that $out_i(s') > out_i(s)$. Therefore, there exists no agent i with a deviation strategy F_i such that $out_i(F_i(s)) > out_i(s)$, so s is an FPE. □

7.2.2 n-person Prisoner's Dilemma

We demonstrate our solution concept on a well-known benchmark example, the n-person Prisoner's Dilemma [SM05].

As we saw in Example 7.1, the Prisoner's Dilemma has two farsighted pre-equilibria: the Nash equilibrium profile where everybody defects, and the "intuitive" solution where everybody cooperates. This extends to the n-person Prisoner's Dilemma.

Definition 7.3 (n-person Prisoner's Dilemma)**.** *Let* $N = \{1,2,\ldots,n\}$ *be the set of agents. Each agent has two strategies: C (cooperate) and D (defect). The payoff function of agent* i *is defined as* $out_i(s_1,\ldots,s_n) = f_i(s_i,h)$ *where* h *is the number of agents other than* i *who play* C *in* s, *and* f_i *is a function with the following properties:*

1. $f_i(D,h) > f_i(C,h)$ *for all* $h = 0,1,\ldots,n-1$;
2. $f_i(C,n-1) > f_i(D,0)$;
3. $f_i(C,h)$ *and* $f_i(D,h)$ *are increasing in* h.

The first requirement states that defecting is always better than cooperating, assuming that the other agents do not change their strategy. The second requirement specifies that the situation where everyone cooperates is better than the situation where everyone defects. The third requirement says that the payoff increases for an agent when a larger number of the other agents cooperate.

Theorem 7.2. *If G is a n-person Prisoner's Dilemma, then the strategy profiles $(C, \ldots, C)$ and $(D, \ldots, D)$ are FPEs in G.*

Proof. We define $s_i^{X_i X_{-i}}$ as $(X_1, \ldots, X_n)$ where $X_j = X_i$ $(1 \leq j \leq n)$ if and only if $j = i$ and $X_j = X_{-i}$ otherwise. Note that $(C, \ldots, C) = s_i^{CC}$ and $(D, \ldots, D) = s_i^{DD}$.

First we show that s_i^{CC} is an FPE. We need to prove that for any deviation strategy F_i, it does not hold that both 1) $out_i(F_i(s_i^{CC})) \geq out_i(s_i^{CC})$ and 2) for every finite F_i-compatible sequence of locally rational deviations $s_i^{CC} = s_1 \to \ldots \to s_k$ we have $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s_i^{CC})$. It does not hold that $out_i(s_i^{CC}) > out_i(s_i^{CC})$, so because of 1), F_i chooses D in s_i^{CC}, i.e. $F_i(\ldots, s_i^{CC}) = s_i^{DC}$. Now consider the F_i-compatible sequence of deviations $s_j^{CC} \to_j s_j^{DC} \to_{-j} \ldots \to_{-j} s_j^{DD}$. This sequence consists of locally rational deviations by assumption 1 in Definition 7.3. Furthermore, either $F_i(\ldots, s_i^{DD}) = s_i^{DD}$ or $F_i(\ldots, s_i^{DD}) = s_i^{CD}$. If $F_i(\ldots, s_i^{DD}) = s_i^{DD}$, then $out_i(s_i^{DD}) < out_i(s_i^{CC})$ by assumption 2, so it does not hold that $out_i(F_i(\ldots, s_i^{DD})) \geq out_i(s_i^{CC})$, which contradicts 2). Furthermore, $out_i(s_i^{CD}) < out_i(s_i^{DD})$ by assumption 1, and because of $out_i(s_i^{DD}) < out_i(s_i^{CC})$, by transitivity $out_i(s_i^{CD}) < out_i(s_i^{CC})$ as well. Therefore, if $F_i(\ldots, s_i^{DD}) = s_i^{CD}$, then it holds that $out_i(F_i(\ldots, s_i^{DD})) < out_i(s_i^{CC})$, which is a contradiction with 2).

Now we show that s_i^{DD} is an FPE for every agent i. It holds that $out_i(s_i^{DD}) \geq out_i(s_i^{CD})$ by assumption 1. Therefore, agent i has no deviation strategy F_i such that $out_i(F_i(s_i^{DD})) > out_i(s_i^{DD})$, so s_i^{DD} is an FPE. □

Example 7.2. *We look at an instance of the 3-person Prisoner's Dilemma. Agent 1 selects rows, agent 2 columns and agent 3 matrices.*

C	C	D
C	$(\mathbf{3}, \mathbf{4}, \mathbf{4})$	$(1, 5, 2)$
D	$(\mathbf{5}, \mathbf{2}, \mathbf{2})$	$(4, 3, 0)$

D	C	D
C	$(1, 2, 5)$	$(0, 3, 3)$
D	$(4, 0, 3)$	$(\mathbf{2}, \mathbf{1}, \mathbf{1})$

The unique Nash equilibrium is (D, D, D), the strategy profile where everyone defects, so this strategy profile is also an FPE. Furthermore, also the strategy profile where everyone cooperates, i.e., (C, C, C), is an FPE. Finally, (D, C, C) is an FPE, showing that also other FPEs can exist.

We can interpret these results as follows. A population where every agent defects might be stable: being the first to cooperate is not necessarily advantageous, as the other agents might not follow. A population where all agents cooperate might also be stable if the agents consider long-term consequences of damaging the opponents' payoffs: if one agent starts defecting, the other agents might follow. Finally, a strategy profile might also be stable if there are only a couple of defecting agents in the population, and the cooperating agents all receive payoffs above some minimal "threshold of fairness" (which is usually the agent's payoff in the Nash equilibrium $(D, \ldots, D)$). Hence the asymmetry: (D, C, C) is farsighted stable, but (C, C, D) and (C, D, C) are not, because they provide agent 1 with an "unfair" payoff, and agent 1 is better off heading for the Nash equilibrium. Another motivation for (D, C, C) to be stable, is that agent 1 does not want to cooperate in the hope that agents 2 and 3 do not change their strategy (as is assumed by Nash equilibrium), while agents 2 and 3 do not want to defect out of fear for follow-ups of the other agents (as is assumed in repeated games).

7.3 Characterizing and Computing FPE

In general, deviation strategies determine the next strategy profile based on the full history of all preceding deviations. In this section, we show that it suffices for the definition of FPE to consider only *positional* deviation strategies, i.e. strategies that determine the next deviation only based on the current strategy profile, independently of what previously happened.

Definition 7.4 (Positional deviation strategy, Positional FPE)**.** *A* positional deviation strategy *for agent i is a strategy F_i such that $F_i(s^1, \ldots, s^k) = F_i(t^1, \ldots, t^k)$ whenever $s^k = t^k$. We will sometimes write $F_i(s^k)$ instead of $F_i(s^1, \ldots, s^k)$ for such strategies. A* positional FPE *is an FPE restricted to positional deviation strategies.*

Theorem 7.3. *A strategy profile $s \in \Sigma$ is an FPE if and only if it is a positional FPE.*

Proof. It suffices to prove that there is no agent i with a deviation strategy such that conditions 1) and 2) from Definition 7.2 hold if and only if there is no agent i with a *positional* deviation strategy such that these conditions hold. Every positional deviation strategy is a deviation strategy, so the 'only if' direction is trivial. We prove the 'if' direction by contraposition. Assume there exists an agent i with a deviation strategy such that conditions 1) and 2) from Definition 7.2 hold in s. Now we define a positional deviation strategy F' as follows. For all $s' \in \Sigma$ for which there exist finite F_i-compatible sequences of locally rational deviations $F_i(s) = s^1 \to \ldots \to s^k = s'$, let $F_i(s) = t^1 \to \ldots \to t^k = s'$ be a shortest F_i-compatible sequence of locally rational deviations. Then we set $F'(s^k) = F(s^0, s^1, \ldots, s^k)$. For all other $s' \in \Sigma$, we set $F'_i(s') = s'$. The function F' is clearly positional and a deviation strategy. Because $F'_i(s)$ is defined based on the shortest sequence which is s (the only sequence of length 1), $F_i(s) = F'_i(s)$, and since we assumed that condition 1) holds for F_i, it also holds for F'_i. Finally we need to check that condition 2) holds. Assume $F'_i(s) = s^1 \to \ldots \to s^k$ is a finite F'_i-compatible sequence of locally rational deviations. Then by definition of F'_i, there exists also a finite F_i-compatible sequence of locally rational deviations $F_i(s) = t^1 \to \ldots \to t^{k-1} \to s^k$ with $F_i(t^1, \ldots, t^{k-1}, s^k) = F'_i(s^k)$. By assumption, $out_i(F_i(t^1, \ldots, t^{k-1}, s^k)) \geq out_i(s)$, so also $out_i(F'_i(s^k)) \geq out_i(s)$. □

The following theorem provides an alternative characterization of farsighted play.

Theorem 7.4. *L is the set of FPEs if and only if for all $s \in L$, all $i \in N$ and all positional deviation strategies F_i with $F_i(s) \neq s$, there exists a finite F_i-compatible sequence of locally rational deviations $s \to_i s^1 \to \ldots \to s^k$ such that $out_i(F_i(s^1, \ldots, s^k)) < out_i(s)$.*

Proof. By Definition 7.2, a strategy profile s is an FPE if and only if there is no agent i with a deviation strategy F_i such that: 1) $out_i(F_i(s)) > out_i(s)$, and 2) for every finite F_i-compatible sequence of locally rational deviations $F_i(s) = s^1 \to \ldots \to s^k$ we have $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$. Because F_i is a deviation strategy, condition 1) is equivalent to $F_i(s) \neq s$ by Definition 7.1. By using this equivalence and moving the negation inwards, we find that a strategy profile s is

an FPE if and only if for every agent i and all deviation strategies F_i such that $F_i(s) \neq s$, there exists a finite F_i-compatible sequence of locally rational deviations $s = s^1 \to \ldots \to s^k$ such that $out_i(F_i(s^1, \ldots, s^k)) < out_i(s)$. By Theorem 7.3, the theorem follows. □

Now we will present a procedure that checks if the strategy profile s is a farsighted pre-equilibrium in game G. Procedure $dev(G, i, s)$ returns *yes* if agent i has a successful deviation strategy from s in G, and *no* otherwise:

1. **forall** $j \in N$ **do** compute relation $\prec_j \in \Sigma \times \Sigma$ st. $t \prec_j t'$ if and only if $\exists t \to_j t'.\ out_j(t) < out_j(t')$;
2. let $\prec_{-i} := \bigcup_{j \neq i} \prec_j$ and let $\ll^*$ be the transitive closure of $\prec_{-i}$;
3. let $Good := \{t \mid out_i(t) \geq out_i(s)\}$; /profiles at least as good as s/
4. **repeat**
   $Good' := Good$;
   forall $t \in Good$ **do**
   if $\exists t' \gg^* t.\ (t' \notin Good \wedge \forall t' \to_i t''.\ t'' \notin Good)$ **then** remove t from $Good'$;
   until $Good' = Good$;
5. **if** $\exists t \in Good.\ s \prec_i t$ **then** return *yes* **else** return *no*.

A strategy profile s is an FPE in G if and only if $dev(G, i, s) = no$ for all agents $i \in N$. Note that the procedure implements a standard greatest fixpoint for a monotonic transformer of state sets. As a consequence, we get that checking if s is a farsighted pre-equilibrium in G can be done in polynomial time with respect to the number of agents and strategy profiles in G.

7.4 Deviations as a Game

Deviations can be seen as moves in a "meta-game" called the *deviation game*, that uses the original payoff matrix as arena. Transitions in the arena (i.e., agents' moves in the meta-game) are given by domination relations of the respective agents. In such a setting, *deviation strategies* can be seen as strategies in the deviation game. A successful deviation strategy for agent i is one that gets i a higher payoff immediately (like in the case of Nash equilibrium), but also guarantees that i's payoff will not drop below the original level after possible counteractions of the opponents. A node in the original game is an FPE exactly when no agent has a winning strategy in the deviation game.

7.4.1 Deviation Games

We formally model deviation games as extensive-form games (e.g. [OR94]). In this chapter, we define *extensive-form games* as follows.

Definition 7.5 (Extensive-form games, finite extensive-form games)**.** *An* extensive-form game *is a tuple* $(H, \mathbb{A}\mathrm{gt}, P, out_1, \ldots, out_n)$ *with the following components:*

- *We call a set of sequences H* prefix-closed *if $\epsilon \in H$, and if $(a^1, \ldots, a^k, a^{k+1}) \in H$ then also $(a^1, \ldots, a^k, \ldots) \in H$. Each member of H is a* history*; each component of a history is an* action *taken by an agent. A history $a^1, \ldots, a^k$ is* terminal *if there is no a^{k+1} such that $a^1, \ldots, a^k, a^{k+1} \in H$. The set of terminal histories is denoted Z. If $A = (a^1, \ldots, a^k)$, we write $A : b$ for $A = (a^1, \ldots, a^k, b)$. We write $b \in (a^1, \ldots, a^k)$ if $b = a^j$ for some $j \in 1, \ldots, k$.*

- *A set of* agents $\mathbb{A}\mathrm{gt}$*;*

- *An* agent *function $P\colon H \backslash Z \to \mathbb{A}\mathrm{gt}$ that assigns to every non-terminal history an agent;*

- *An outcome function $out\colon Z \to \mathbb{R}$ that assigns to every terminal history an outcome.*

We say that an extensive-form game is finite *if every history is finite.*

A deviation game D is constructed from a strategic game G and a strategy profile s in G, and consists of two phases. In the first phase, each agent can either start deviating from s or pass the turn to the next agent. If no agent deviates, all agents get the "neutral" payoff 0 in D. If an agent i deviates, the game proceeds to the second phase in which i tries to ensure that his deviation is successful, while all other agents try to prevent it. This phase is strictly competitive: if i succeeds, he gets the payoff of 1 and all the other agents get -1; if i fails, he gets -1 and the other agents get 1 each.

Formally, given a strategic game $G = (N, \Sigma_1, \ldots, \Sigma_n, out_1, \ldots, out_n)$ and a strategy profile s, the deviation game is an extensive-form game

$$T(G, s) = (N, H, P, out'_1, \ldots, out'_n),$$

where N is the set of agents as in G, H is the set of histories in the deviation game, P is a function assigning an agent to every non-terminal history, and for every $i \in N$, out'_i is a function assigning the payoff for agent i to every terminal history. A history in H is a sequence of nodes of the form (i, t, j), with the intended meaning that $i \in N \cup \{-\}$ is the agent whose deviation strategy is currently tested (where "$-$" means that no deviation has been made yet), $t \in \Sigma$ is the current strategy profile under consideration, and $j \in N \cup \{\perp\}$ is the agent currently going to play (i.e., $P(\ldots, (i, t, j)) = j$), where $\perp$ indicates that the game has terminated. The initial state is $(-, s, 1)$. For every agent j, we define out'_j as follows:

- $out'_j(\ldots, (-, t, \perp)) = 0$;

- if $out_i(t) \geq out_i(s)$, then $out'_j(\ldots, (i, t, \perp)) = 1$ when $j = i$, otherwise $out'_j(\ldots, (i, t, \perp)) = -1$;

- if $out_i(t) < out_i(s)$, then $out'_j(\ldots, (i, t, \perp)) = -1$ when $j = i$, otherwise $out'_j(\ldots, (i, t, \perp)) = 1$.

Now we recursively define the set of histories H. We will write $\underline{i}$ for $\min(N \backslash \{i\})$.

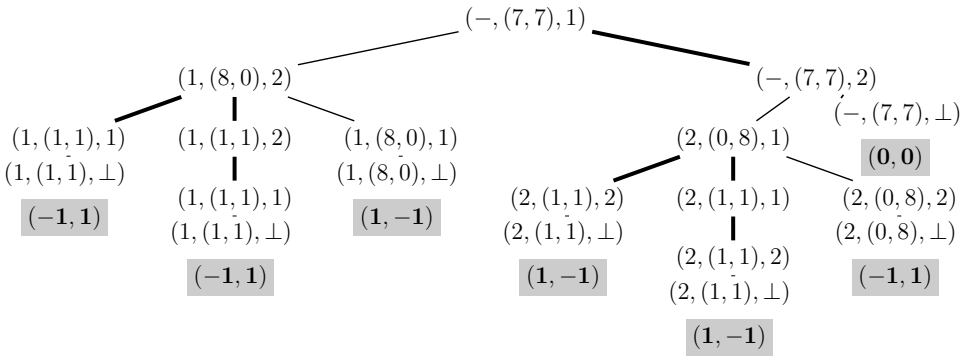


Figure 7.1: Deviation game for strategy profile $(7, 7)$ in Prisoner's Dilemma (Example 7.1 (A))

1. $\epsilon \in H$, where ϵ is the empty sequence.
2. $(-, s, 1) \in H$.
3. If $h = \ldots, (-, s, i) \in H$ and $i + 1 \in N$ then $h, (-, s, i + 1) \in H$.
4. If $h = \ldots, (-, s, i) \in H$ and $i = \max(N)$, then $h, (-, s, \bot) \in H$.
5. If $h = \ldots, (-, s, i) \in H$, $s \to_i s'$ is a locally rational deviation and $\underline{i} \in N \backslash \{i\}$ then $h, (i, s', \underline{i}) \in H$.
6. If $h = \ldots, (i, s', i) \in H$ and $s' \to_i s''$ is a locally rational deviation, then $h, (i, s'', \underline{i}) \in H$.
7. If $h = \ldots, (i, s', i) \in H$, then $h, (i, s', \bot) \in H$.
8. If $h = \ldots, (i, s', i') \in H$, $i' \in N \backslash \{i\}$ and either $s' \to_i s''$ is a locally rational deviation or $s' = s''$, then $h, (i, s'', i') \in H$ whenever both $h, (i, s'', i') \notin H$ and $i' = i$ implies $h, (-, s'', i') \notin H$.

Statement 2 specifies the initial history. Statements 3–5 say that if nobody has deviated so far, agent i can embark on a deviation strategy or refrain from deviating and pass the token further. If no agent deviates, the game ends. If agent i initiates deviations, the strategy profile changes, and the token goes to the first opponent. Statement 6 says that the latter also applies during execution of the deviation strategy. Furthermore, 7 indicates that agent i can stop the game if it is his turn (note that this can only be the case if the opponents do not want to deviate anymore). Finally, 8 states that an opponent agent can make a locally rational deviation or do nothing if it is his turn, and pass the turn to another agent i' (as long as the agent has not had the turn in the new strategy profile before, to guarantee finite trees).

Now we can see an *opponent strategy* against agent i as a set of strategies for agents $N \backslash \{i\}$ such that every deviation is locally rational, and in every strategy profile, not more than one agent deviates. Formally, an opponent strategy against agent i is a function $F_{-i} : N \backslash \{i\} \times \Sigma^* \to \Sigma$ such that for every agent $j \in N \backslash \{i\}$, $s \to_j F_{-i}(j, (\ldots, s))$ is a locally rational deviation or $F_{-i}(j, (\ldots, s)) = s$ and such that $F_{-i}(j, (\ldots, s)) \neq s$ for some j implies $F_{-i}(j', (\ldots, s)) = s$ for all $j' \neq j$.

Example 7.3. *Figure 7.1 depicts the deviation game $T(G, s)$ where G is the Prisoner's Dilemma and s is $(7, 7)$. The moves selected by the minimax algorithm (e,g,[OR94]) are printed as thick lines. The minimax algorithm selects outcome $(0, 0)$, so no agent has a strategy yielding more than 0, which indicates that $(7, 7)$ is an FPE.*

7.4.2 Correspondence to FPE

Now we will prove that a strategy profile in the original game is an FPE exactly when no agent has a strategy that guarantees the payoff of 1 in the deviation game. We say that a sequence of strategy profiles $s^1, \ldots, s^k$ is (F_i, F_{-i})-*compatible* if for all $k' < k$, either $F_{-i}(j, (s^1, \ldots, s^{k'})) = s^{k'+1}$ for some $j \in N \backslash \{i\}$, or both $F_i(s^1, \ldots, s^{k'}) = s^{k'+1}$ and $F_{-i}(j, (s^1, \ldots, s^{k'})) = s^{k'}$ for all $j \in N \backslash \{i\}$. Furthermore, a sequence of strategy profiles $s^1, \ldots, s^k$ is *loop-free* if $s^n \neq s^{n'}$ for $1 \leq n \leq n' \leq k$.

Let G be a strategic form game, s be a strategy profile and $i \in N$ be an agent. Now we say that a deviation strategy F_i is successful against an opponent strategy F_{-i}, if 1) $out_i(F_i(s)) > out_i(s)$, and 2) for every loop-free (F_i, F_{-i})-compatible sequence of strategy profiles $F_i(s) = s^1, \ldots, s^k$, it holds that $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$. The following lemma shows that it is indeed sufficient to look at loop-free (F_i, F_{-i})-compatible sequences.

Lemma 7.1. *Strategy profile s is an FPE in game G if and only if there does not exist an agent i with a deviation strategy F_i that is successful against all opponent strategies F_{-i}.*

Proof. First we prove the 'only if' direction by contraposition. Assume there exists an agent i with a deviation strategy F_i that is successful against all opponent strategies F_{-i}, i.e., 1) $out_i(F_i(s)) > out_i(s)$, and 2) for every loop-free (F_i, F_{-i})-compatible sequence of strategy profiles $F_i(s) = s^1, \ldots, s^k$, it holds that $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$.

Let $F_i(s) = s^1 \to \ldots \to s^k$ be a loop-free F_i-compatible sequence of locally rational deviations. We define opponent strategy F_{-i} such that $F_{-i}(j, s^k) = s^{k+1}$ whenever $s^k \to_j s^{k+1}$ for $j \in N \backslash \{i\}$. Then $s^1, \ldots, s^k$ is (F_i, F_{-i})-compatible, so we have $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$ by assumption. Therefore, for every loop-free F_i-compatible sequence of locally rational deviations $F_i(s) = s^1 \to \ldots \to s^k$, it holds that $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$ $(*)$.

Now let $F_i(s) = s^1 \to \ldots \to s^k$ be a finite F_i-compatible sequence of locally rational deviations. Then we can construct a loop-free sequence $t^1, \ldots, t^k$ with $t_1 = s_1$ and $t_k = s_k$. Now $out_i(F_i(t^1, \ldots, t^k)) = out_i(F_i(s^k))$ because F_i is positional, and $out_i(F_i(t^1, \ldots, T^k)) \geq out_i(s)$ by $(*)$. Therefore, for every finite F_i-compatible sequence of locally rational deviations $F_i(s) = s^1 \to \ldots \to s^k$, it holds that $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$. This shows that there exists an agent i with a deviation strategy F_i such that 1) $out_i(F_i(s)) > out_i(s)$, and 2) for every finite F_i-compatible sequence of locally rational deviations $F_i(s) = s^1 \to \ldots \to s^k$, it holds that $out_i(F_i(s^1, \ldots, s^k)) \geq out_i(s)$, i.e., s is not an FPE.

The 'if' direction follows from the fact that every loop-free sequence of strategy

profiles is finite, and the fact that when a sequence of strategy profiles is (F_i,F_{-i})-compatible, it is also an F_i-compatible sequence of deviations. □

We proceed by defining a bijection ϕ between strategy F_i in G and strategy Φ_i in $T(G,s)$ as follows.

If $F_i(s) = s$ then $\Phi_i(-,s,i) = (-,s,i+1)$ where $i+1 \in N$;
If $F_i(s) = s$ then $\Phi_i(-,s,i) = (-,s,\bot)$ where $i = \max(N)$;
If $F_i(s) = s'$ then $\Phi_i(-,s,i) = (i,s',\underline{i})$ where $s \neq s'$;
If $F_i(s') = s'$ then $\Phi_i(i,s',i) = (i,s',\bot)$; where $s \neq s'$;
If $F_i(s') = s''$ then $\Phi_i(i,s',i) = (i,s'',\underline{i})$ where $s \neq s' \neq s''$.

We call a set of strategies Φ_{-i} for agents $N\backslash\{i\}$ *non-initially-deviating* whenever $\Phi_{i'}(-,s,i') = (-,s,i'')$ where $i \neq i'$. Then an *opponent strategy* Φ_{-i} in the deviation game is a set of non-initially-deviating strategies Φ_j for agents $j \in N\backslash\{i\}$ such that in every strategy profile, not more than one agent in $N\backslash\{i\}$ deviates and the other agents always give the turn to the deviating agent, i.e., $\Phi_j(i,s',j) = (i,s'',j')$ with $s' \neq s''$ for some $j,j' \in N\backslash\{i\}$ implies $\Phi_i(i,s',j'') = (i,s',j)$ for all $j'' \neq j$. Now we define a bijection ψ between an opponent strategy F_{-i} in T and an opponent strategy Φ_{-i} in $T(G,s)$. Let $\psi(F_{-i}) = \Phi_{-i}$, where Φ_{-i} is defined as follows:

- If $F_{-i}(i',s') \neq s'$ for some $i' \in N\backslash\{i\}$, then $\Phi_{i'}(i,s',i') = (i,s'',i')$ and $\Phi_{i''}(i,s',i'') = (i,s',i')$ for $i'' \neq i'$.
- If $F_{-i}(i',s') = s'$ for all $i' \in N\backslash\{i\}$, then $\Phi_{i'}(i,s'',i') = (i,s'',i)$.

It can easily be checked that ϕ and ψ are indeed bijections.

Let $out_i(\Phi_i,\Phi_{-i})$ be the outcome of the game for agent i when agent i plays strategy Φ_i and agents $N\backslash\{i\}$ play strategy Φ_{-i}. When $out_i(\Phi_i,\Phi_{-i}) = u_i$ and $out_j(\Phi_i,\Phi_{-i}) = u_{-i}$ for $j \in N\backslash\{i\}$, we sometimes write $out_{i,-i}(\Phi_i,\Phi_{-i}) = (u_i,u_{-i})$.

Lemma 7.2. *If $i \in N$ is an agent with a deviation strategy F_i and F_{-i} is an opponent strategy, then F_i is successful against F_{-i} in game G and strategy profile s if and only if $out_{i,-i}(\phi(F_i),\psi(F_{-i})) = (1,-1)$ in $T(G,s)$.*

Proof. By construction of ψ and ϕ, we have a run $s = s^1 \to_{i^1} s^2 \to_{i^2} \ldots \to_{i^{k-1}} s^k$ in G if and only if

$$(-,s,1),\ldots,(-,s,i),(i,s',\underline{i}),\ldots,(i,s^k,i),(i,F_i(s^k),\underline{i}),(i,F_i(s^k),i),(i,F_i(s^k),\bot)$$

is a run in $T(G,s)$. Therefore, a run ends in s^k in G with $out_i(F_i(s^1,\ldots,s^k)) \geq out_i(s)$ if and only if a run ends in $(i,F_i(s^k),\bot)$ in $T(G,s)$ with $out_i(F_i(s^k)) \geq out_i(s)$. Therefore, F_i is successful against F_{-i} if and only if $out_i(\phi(F_i),\psi(F_{-i})) = 1$. □

Theorem 7.5. *Strategy profile $s \in \Sigma$ is an FPE in game G if and only if all subgame-perfect Nash equilibria in $T(G,s)$ yield $(0,\ldots,0)$.*

Proof. To prove the 'only if' direction, we assume that strategy profile s is an FPE in game G. By Lemma 7.1, there does not exist an agent i with a deviation strategy F_i that is successful against all opponent strategies F_{-i}. Because f

and g are bijections, by Lemma 7.2 there does not exist an agent i with a strategy Φ_i such that for every opponent strategy Φ_{-i} it holds that $out_{i,-i}(\Phi_i, \Phi_{-i}) = (1, -1)$. This means that for every agent i with a strategy Φ_i, there exists an opponent strategy Φ_{-i} such that $out_{i,-i}(\Phi_i, \Phi_{-i}) \neq (1, -1)$ $(*)$. Now we prove that every *subgame-perfect Nash equilibrium* (*SPNE*) (e.g. [OR94]) $(\Phi_1, \ldots, \Phi_n)$ in the subgame starting at $(-, s, i)$ yields $(0, \ldots, 0)$ by backwards induction on $i \in (1, \ldots, n, \bot)$. The base case, where $i = \bot$, follows from the definition of *out*. Now assume that the claim holds for $i + 1$ (where $n + 1 = \bot$). To show that $\Phi_i(-, s, i) = (i, s, i + 1)$, we assume that $\Phi_i(-, s, i) = (i, s', \underline{i})$ for some s' and derive a contradiction. Let Φ_{-i} be an opponent strategy. Now $out_{i,-i}(\Phi_i, \Phi_{-i})$ is either $(1, -1)$ or $(-1, 1)$. If $out_{i,-i}(\Phi_i, \Phi_{-i}) = (1, -1)$, by $(*)$, there exists an opponent strategy Φ'_{-i} such that $out_{i,-i}(\Phi_i, \Phi'_{-i}) \neq (1, -1)$ and thus $out_{i,-i}(\Phi_i, \Phi'_{-i}) = (-1, 1)$. Now $out_{-i}(\Phi_i, \Phi'_{-i}) > out_{-i}(\Phi_i, \Phi_{-i})$, which contradicts the assumption that (Φ_i, Φ_{-i}) is a Nash equilibrium. If $out_{i,-i}(\Phi_i, \Phi_{-i}) = (-1, 1)$, let Φ'_i be a strategy such that $\Phi'_i(-, s, n) = (-, s, n+1)$ and Φ'_i is a SPNE strategy in the subgame starting at $(-, s, n + 1)$. Then $out_i(\Phi'_i, \Phi'_{-i}) \geq 0$ for all opponent strategies Φ'_{-i} by the induction hypothesis This implies that $out_i(\Phi'_i, \Phi_{-i}) > out_i(\Phi_i, \Phi_{-i})$, contradicting the assumption that (Φ_i, Φ_{-i}) is a Nash equilibrium. This implies that the assumption $\Phi_i(-, s, i) = (i, s', \underline{i})$ is false, so $\Phi_i(-, s, i) = (i, s, i + 1)$ or $\Phi_i(-, s, i) = (i, s, \bot)$. By the induction hypothesis, all SPNE in the subgame starting at $(-, s, i + 1)$ yield $(0, \ldots, 0)$. Therefore, all SPNE in $T(G, s)$ yield $(0, \ldots, 0)$.

We prove the 'if' direction by contraposition. Assume strategy profile $s \in \Sigma$ is not an FPE in game G. By Lemma 7.1, there exists an agent i with a strategy F_i that is successful against all opponent strategies F_{-i}. Because ϕ and ψ are bijections, by Lemma 7.2 there exists an agent i with a strategy Φ_i such that for every opponent strategy Φ_{-i} it holds that $out_{i,-i}(\Phi_i, \Phi_{-i}) = (1, -1)$ $(\dagger)$. Now let (Φ'_i, Φ'_{-i}) be a strategy profile such that $out_i(\Phi'_i, \Phi'_{-i}) = 0$. Then there exists a strategy Φ_i such that $out_i(\Phi_i, \Phi'_{-i}) = 1$ by $(\dagger)$, so $out_i(\Phi_i, \Phi'_{-i}) > out_i(\Phi'_i, \Phi'_{-i})$, and therefore (Φ'_i, Φ'_{-i}) is not an SPNE. An extensive game always has an SPNE and (Φ'_i, Φ'_{-i}) is the only strategy profile yielding $(0, \ldots, 0)$, so there exist SPNEs not yielding $(0, \ldots, 0)$, which implies that not all SPNEs yield $(0, \ldots, 0)$. □

Note that Theorem 7.5 provides an alternative way of checking pre-equilibria: s is an FPE in G if and only if the minimaxing algorithm [OR94] on $T(G, s)$ returns 0 for every agent. However, the deviation game for G can be exponentially larger than G itself, so the algorithm proposed in Section 7.3 is more efficient.

7.5 Comparing Farsighted Solution Concepts

There have been a number of solution concepts with similar agenda to FPE. In this section, we discuss how they compare to our new proposal.

7.5.1 Related Work

The discussion on myopic versus farsighted play dates back to the *von Neumann-Morgenstern stable set* (VNM) in coalitional games [NM44], and *indirect dominance in Harsanyi's sense* of coalition structures, leading to the *strictly stable set*

(SSS) [Har74]. More recent proposals are the *noncooperative farsighted stable set* (NFSS) [Nak07] and the *largest consistent set* (LCS) [Chw94]. Other similar solution concepts include [Gre90, SM05, DX03]. Also Halpern and Rong's *cooperative equilibrium* [HR10] can be seen as a farsighted solution concept.

Definitions In order to define the concepts, we introduce three different dominance relations between strategy profiles. *Direct dominance* of x over y means that agent i can increase his own payoff by deviating from strategy profile x to strategy profile y. *Indirect dominance* of x over y says that a coalition of agents can deviate from strategy profile x to strategy profile y, possibly via a number of intermediate strategy profiles, such that every coalition member's final payoff is better than his payoff before his move. Finally, *indirect dominance in Harsanyi's sense* is indirect dominance with the additional requirement that each individual deviation is locally rational. Formally:

- We say that y directly dominates x through agent i ($x \prec_i y$) if there is a locally rational deviation $x \to_i y$. We also write $x \prec y$ if $x \prec_i y$ for some $i \in N$.
- We say that y indirectly dominates x ($x \ll y$) if there exists a sequence of (not necessarily locally rational) deviations $x = x^0 \to_{i_1} x^1 \ldots \to_{i_p} x^p = y$ such that $out_{i_r}(x^{r-1}) < out_{i_r}(y)$ for all $r = 1, 2, \ldots, p$.
- We say that x indirectly dominates y in Harsanyi's sense ($x \ll_{\mathrm{H}} y$) if there exists a sequence of locally rational deviations $x = x^0 \to_{i_1} x^1 \ldots \to_{i_p} x^p = y$ such that $out_{i_r}(x^{r-1}) < out_{i_r}(y)$ for all $r = 1, 2, \ldots, p$.

It can easily be seen that $x \prec y$ implies $x \ll_{\mathrm{H}} y$, and $x \ll_{\mathrm{H}} y$ implies $x \ll y$.

Example 7.4. *In the Prisoner's Dilemma (Example 7.1(A)), we have* $(7,7) \prec_1 (8,0)$, $(0,8) \prec_1 (1,1)$, $(7,7) \prec_2 (0,8)$ *and* $(8,0) \prec_2 (1,1)$. *In addition,* $(7,7)$ *indirectly dominates* $(1,1)$ *in Harsanyi's sense, i.e.,* $(1,1) \ll_{\mathrm{H}} (7,7)$.

With these definitions, we can introduce four main farsighted solution concepts.

- A subset K of Σ is a *von Neumann-Morgenstern stable set* (VNM) if it satisfies the following two conditions: (a) for all $x, y \in K$, neither $x \prec y$ nor $y \prec x$; (b) for all $x \in \Sigma \backslash K$, there exists $x \in K$ such that $x \prec y$ [NM44]. In fact, a VNM corresponds to stable extensions in the argumentation theory $(\Sigma, \prec')$, where $\prec'$ is the converse of $\prec$, in Dung's argumentation framework [Dun95].
- Replacing the direct dominance relation $\prec$ in VNM by the indirect dominance relation $\ll$, yields the *noncooperative farsighted stable set* (NFSS) [Nak07].
- Furthermore, a subset S of Σ is a *strictly stable set* (SSS) if it is a VNM such that for all $x, y \in S$, neither $x \ll_{\mathrm{H}} y$, nor $y \ll_{\mathrm{H}} x$ [Har74].
- Finally, a subset L of Σ is consistent in Chwe's sense if ($x \in L$ if and only if for all deviations $x \to_i y$ there exists $z \in L$ such that [$y = z$ or $y \ll z$] and $out_i(x) \geq out_i(z)$). Now the *largest consistent set* (LCS) is the union of all the consistent sets in Σ [Chw94].

	L	R
T	$(3,1)$	$(\mathbf{0},\mathbf{2})$
B	$(\mathbf{1},\mathbf{1})$	$(0,0)$

(a)

F	L	R
T	$(3,1,2)$	$(\mathbf{0},\mathbf{2},\mathbf{2})$
B	$(0,0,0)$	$(0,0,0)$

S	L	R
T	$(3,1,1)$	$(0,0,0)$
B	$(\mathbf{1},\mathbf{1},\mathbf{1})$	$(0,0,0)$

(b)

	L	R
T	$(3,1)$	$(\mathbf{2},\mathbf{2})$
B	$(1,1)$	$(0,0)$

(c)

	L	C	R
T	(0,0)	$(3,1)$	$(\mathbf{0},\mathbf{2})$
B	$(\mathbf{1},\mathbf{3})$	$(\mathbf{2},\mathbf{4})$	$(0,0)$

(d)

Figure 7.2: Example games (FPEs are printed in bold)

Another solution concept that has been recently proposed is *perfect cooperative equilibrium* (PCE) [HR10]. Like FPE, PCE aims at explaining situations where cooperation is observed in practice. An agent's payoff in a PCE is at least as high as in any Nash equilibrium. However, a PCE does not always exist. Every game has a Pareto optimal maximum PCE (M-PCE), as defined below. We only give the definition for 2-agent games; the definition for n-person games can be found in [HR10].

Given a game G, a strategy s_i for agent i in G is a best response to a strategy s_{-i} for the agents in $N\backslash\{i\}$ if $U_i(s_i, s_{-i})) = \sup_{s'_i \in \Sigma_i} U_i(s'_i, s_{-i})$. Let $BR_i(s_{-i})$ be the set of best responses to s_{-i}. Given a 2-agent game, let BU_i denote the best payoff that agent i can obtain if the other agent j best responds, that is $BU_i = \sup_{s_i \in \Sigma_i, s_j \in BR(s_i)} U_i(s)$. A strategy profile is a PCE if for $i \in \{1,2\}$ we have $U_i(s) \geq BU_i$. A strategy profile is an α-PCE if $U_i(s) \geq \alpha + BU_i$ for all $i \in N$. The strategy profile s is an M-PCE if s is an α-PCE and for all $\alpha' > \alpha$, there is no α'-PCE.

Example 7.5. *In the Prisoner's Dilemma (Example 7.1(A)), there is one VNM* ($\{(1,1),(7,7)\}$) *and one NFSS* ($\{(7,7)\}$). *There is no SSS. Finally, the LCS is* $\{(1,1),(7,7)\}$.

Regarding PCE, we have $BR_1(C) = \{D\}$, $BR_1(D) = \{D\}$, $BR_2(C) = \{D\}$, *and* $BR_2(D) = \{D\}$. *This implies that* $BU_1 = D$ *and* $BU_2 = D$. *Thus, the set of PCE outcomes is* $\{(7,7),(1,1)\}$, *and* $(7,7)$ *is the unique M-PCE (with* $\alpha = 6$*).*

7.5.2 FPE vs. Other Farsighted Concepts

The main idea of all introduced farsighted solution concepts (except PCE) is very similar. One can test whether a given strategy profile is stable by checking whether an agent or group of agents can deviate from the strategy profile in a profitable way, given a possible follow-up from the other agents. Strategy profiles from which no agent has an incentive to deviate, after taking the follow-up by the other agents in account, are considered stable. However, there are also many differences between the concepts. In this section, we will compare FPE with other farsighted solution concepts in various aspects.

Scope of Farsightedness In farsighted reasoning about strategies, agents consider further consequences of their deviations, as opposed to reasoning in myopic solution concepts like Nash equilibrium. Consider for example the game in Figure 7.2a. Strategy profile $(1,1)$ is not a Nash equilibrium because $(1,1) \rightarrow_1 (3,1)$

is locally rational. However, this deviation is not necessarily *globally* rational, as it might trigger agent 2 to follow up with the deviation $(3,1) \to_2 (0,2)$. Unlike Nash equilibrium, which only considers (0,2) stable, the set $\{(1,1),(0,2)\}$ is considered stable in all presented farsighted solution concepts (VNM, NFSS, SSS, LCS and FPE).

The degree of farsightedness is different across the concepts. The least farsighted concept is VNM. Here, only the direct dominance relation $\prec$ is taken into account. This implies that agents only look at whether they can recover from a *single* deviation of the opponents, as the game in Figure 7.2b illustrates. The deviation $(1,1,1) \to_1 (3,1,1)$ is locally rational but might intuitively be wrong because deviations $(3,1,1) \to_3 (3,1,2) \to_2 (0,2,2)$ can spoil its effect. However, since VNM does not take sequences of deviations into account, it does not consider $(1,1,1)$ stable ($\{(0,2,2),(3,1,1)\}$ being the only stable set). The concepts NFSS, LCS and FPE have a "more farsighted" view, and consider sequences of follow-up deviations. In consequence, they all deem the profile $(1,1,1)$ stable.

Furthermore, the solution concepts evaluate follow-ups differently. In VNM and NFSS, a follow-up deviation from the opponents is always considered undesirable, even if it gives a higher payoff for the first deviating agent. In LCS and FPE, beneficial follow-ups only strengthen the success of the original deviation. Consider the game in Figure 7.2c. After $(1,1) \to_1 (3,1)$, the follow-up $(3,1) \to_2 (2,2)$ still leaves agent 1 with a payoff higher that his initial one. Thus, both LCS and FPE deem $(1,1)$ unstable, which matches intuition, while VNM and NFSS consider $(1,1)$ stable.

Type of Solution Concept The concepts also yield objects of different types. LCS and FPE both return a set of strategy profiles, thus ascribing rationality to *individual profiles*. On the other hand, VNM, NFSS and SSS return a set of sets of profiles each, hence ascribing rationality to *sets* of strategy profiles. In the latter case a rational set of profiles can be understood as a set of collective decisions to whom the grand coalition of agents can consistently stick. Clearly, this makes sense in coalitional games, but is less suitable for noncooperative games where the agents' control over collective choice is limited. Furthermore, still no prediction is made about which set from the collection will be selected by the agents to be stable.

Deviation Strategy VNM, SSS and FPE are built on a pessimistic view of the follow-up to the first deviation, as they make no assumptions about the other agents' rationality. In particular, it is not assumed that opponents will help to increase the initiator's utility, even if it is also to their advantage. In consequence, these solution concepts assume that the deviations of the initiator must always be locally rational. In contrast, NFSS assumes that an agent can make deviations which are not locally rational if he hopes that other agents will further increase his utility. The game in Figure 7.2a illustrates this. The set $\{(1,1),(0,2)\}$ is a VNM, SSS, LCS and collects all FPEs. On the other hand, $\{(0,2)\}$ is the only NFSS. FPE and (M-)PCE also make different assumptions about the initially deviating agent. The intuition of these solution concepts is similar: agents realize that if they deviate from a strategy profile, the other agent may start to best respond; then, after a while, they will likely end up in some Nash equilibrium, and thus have

a payoff that is guaranteed to be no less than that of the original strategy profile. The solution concepts PCE and M-PCE may also require agents to deviate in a locally irrational way because they do not take into account the domination relation explicitly. For example, $(0, 2)$ in Figure 7.2d is neither PCE nor M-PCE, although it is a Nash equilibrium and hence no agent has a locally rational deviation in it. All the other solution concepts considered here deem $(0, 2)$ stable.

Expected Behavior of Opponents Different solution concepts imply different opponent models. We have already mentioned that the initiator of deviations can either be optimistic or pessimistic about the follow-up by the opponents. Another distinction is whether the deviating agent expects the opponents to be farsighted as well, or whether they might be regular best-response agents. Consider the game in Figure 7.2d. Intuitively, a farsighted agent 1 would not deviate $(2, 4) \to_1 (3, 1)$, because the follow-up deviation $(3, 1) \to_2 (0, 2)$ can damage his payoff. Therefore, agent 2 can safely play $(1, 3) \to_2 (2, 4)$ if he is sure that agent 1 is farsighted. However, if agent 1 plays best response, the deviation $(1, 3) \to_2 (2, 4)$ might harm agent 2, because agent 1 will deviate $(2, 4) \to_1 (3, 1)$ afterwards. Therefore, if agent 2 has no information about the kind of behavior of agent 1, it might be better to stick to strategy profile $(1, 3)$. FPE is the only solution concept that captures this intuition by considering $(1, 3)$, $(2, 4)$ and $(0, 2)$ to be (potentially) stable; the other formalisms (VNM, SSS, NFSS, LCS) all result in the stable set $\{(1, 4), (0, 2)\}$.

Summary The main difference between our farsighted pre-equilibrium and the other solution concepts discussed in this section lies in the perspective. It can be argued that the type of rationality defined in [NM44, Har74, Chw94, SM05, Nak07] is predominantly coalitional. This is because those proposals ascribe stability to *sets* of strategy profiles, which does not have a natural interpretation in the noncooperative setting. Moreover, some of the concepts are based on coalitional rather than individual deviations. In contrast, the concept of cooperative equilibrium [HR10] is *not* based on reasoning about possible deviations. In this sense, FPE is the first truly noncooperative solution concept for farsighted play that we are aware of.

7.6 Conclusion

We have proposed a new solution concept that we call *farsighted pre-equilibrium*. The idea is to "broaden" Nash equilibrium in a way that does not discriminate solutions that look intuitively appealing but are ruled out by Nash equilibrium. Then, Nash equilibrium may be interpreted as a specification of play which is certainly rational, and strategy profiles that are *not* farsighted pre-equilibria can be considered certainly *irrational*. The area in between is the gray zone where solutions are either rational or not, depending on the detailed circumstances.

Our main motivation is predictive: we argue that a solution concept that makes too strong assumptions open up ways of possible vulnerability if the other agents do not behave in the predicted way. Nash equilibrium seems too restrictive in many games (Prisoner's Dilemma being a prime example). We show that FPE does select non-Nash equilibrium strategy profiles that seem sensible, like the "all cooperate" strategy profile in the standard as well as the generalized version of Prisoner's Dilemma. Moreover, we observe that FPE favors solutions with bal-

anced distributions of payoffs, i.e., ones in which no agent has significantly higher incentive to deviate than the others.

A natural way of interpreting deviations in strategy profiles is to view the deviations as moves in a "deviation game" played on a meta-level. We show that farsighted pre-equilibria in the original game correspond to subgame-perfect Nash equilibria in the meta-game. This is a strong indication that the concept that we propose is well-rooted in game-theoretic tradition of reasoning about strategic choice.

Farsighted play has been investigated in multiple settings, starting from von Neumann and Morgenstern almost 70 years ago. Our proposal is, to our knowledge, the first truly noncooperative solution concept for farsighted play. In particular, it is obtained by reasoning about *individual* (meta-)strategies of *individually* rational agents, rather than by reconstruction of the notion of *stable set* from coalitional game theory.

8

Relating Attack-Defense Trees and Games

Abstract. Attack–defense trees are used to describe security weaknesses of a system and possible countermeasures. In this chapter, the connection between attack–defense trees and game theory is made explicit. We show that attack–defense trees and binary zero-sum two-player extensive-form games have equivalent expressive power when considering satisfiability, in the sense that it is possible to convert ADTerms to these games (and vice-versa) while preserving their outcome and their internal structure.

8.1 Introduction

In this thesis, we have seen how security protocols can be modeled as a game between an attacker and a defender. This approach can also be fruitful in other areas, such as security modeling. In this chapter, we model an extension of *attack trees* [Sch04a], called *attack–defense trees* [KMRS10], as a game.

Attack trees [Sch04a], as popularized by Bruce Schneier at the end of the 1990s, form an informal but powerful method to describe possible security weaknesses of a system. An attack tree basically consists of a description of an attacker's goal and its refinement into sub-goals. In case of a *conjunctive* refinement, all sub-goals have to be satisfied to satisfy the overall goal, while for a *disjunctive* refinement satisfying any of the sub-goals is sufficient. The non-refined nodes (i.e., the leaves of the tree) are basic attack actions from which complex attacks are composed.

Due to their intuitive nature, attack trees have proven to be very useful in understanding a system's weaknesses in an informal and interdisciplinary context. The development of an attack tree for a specific system may start by building a small tree that is obviously incomplete and describes the attacks at a high level of abstraction. Later, these attacks might be refined and new attacks might be added later as to make a more complete description.

Over the last few years, attack trees have developed into an even more versatile tool. This is due to two developments. The first development consists of the formalization of the attack trees method [MO05], which provides an attack tree with a precise meaning. As a consequence, formal analysis techniques were designed [JW09, RSF+09], and computer tools were made commercially available [Ame, Iso].

The second development comes from the insight that a more complete description can be achieved by modeling the activities of a system's defender in addition to those of the attacker. Consequently, one can analyze which set of defenses is optimal from the perspective of, for instance, cost effectiveness. Several notions

of protection trees or defense nodes have already been proposed in the literature [EDRM06, BDP06]. They mostly consist of adding one layer of defenses to the attack tree, thus ignoring the fact that in a dynamic system new attacks are mounted against these defenses and that, consequently, yet more defenses are brought into place. Such an alternating nature of attacks and defenses is captured in the notion of attack–defense trees [KMRS10]. In this extension of attack trees, the iterative structure of attacks and defenses can be visualized and evolutionary aspects can be modeled.

These two developments, the formalization of attack trees and the introduction of defenses, imply that an attack–defense tree can be formally considered as a description of a game. The purpose of this chapter is to make the connection between attack–defense trees and game theory explicit. We expect that the link between the relatively new field of attack modeling and the well-developed field of game theory can be exploited by making game theoretic analysis methods available to the attack modeling community. In particular, we study the relation between attack–defense trees and games in terms of expressiveness. Rather than studying the graphical attack–defense tree language, we consider an algebraic representation of such trees, called *attack–defense terms* (*ADTerms*) [KMRS10], which allows for easier formal manipulation.

The main contribution of this chapter is to show that ADTerms with a satisfiability attribute are equivalent to two-agent binary zero-sum extensive-form games. In this chapter, we refer only to this class of games when we talk about games. We show equivalence by defining two translations: one from games to ADTerms and one from ADTerms to games. Then, we interpret a *strategy* in the game as a *basic assignment* for the corresponding ADTerm and vice versa. Such a basic assignment expresses which attacks and defenses are in place. Equivalence then roughly means that for every winning strategy, there exists a basic assignment that yields a satisfiable term, and vice versa. Although the two formalisms have much in common, their equivalence is not immediate. Two notions in the domain of ADTerms have no direct correspondence in the world of games: conjunctive nodes and refinements. The translation from ADTerms to games will have to solve this in a semantically correct way.

This chapter is structured as follows. We introduce attack–defense terms and two-agent binary zero-sum extensive-form games in Section 8.2. In Section 8.3, we define a translation from games to attack–defense terms and prove that an agent can win the game if and only if he is successful in the corresponding ADTerm. A reverse translation is defined in Section 8.4, before we conclude in Section 8.5.

8.2 Preliminaries

8.2.1 Attack–Defense Trees

A limitation of attack trees is that they cannot capture the interaction between attacks carried out on a system and defenses put in place to fend off the attacks. To mitigate this problem and in order to be able to analyze an attack–defense scenario, attack–defense trees are introduced in [KMRS10]. Attack–defense trees may have

two types of nodes: attack nodes and defense nodes, representing actions of two opposing agents. The attacker and defender are modeled in a purely symmetric way. To avoid differentiating between attack–defense scenarios with an attack node as a root and a defense node as a root, the notions of *proponent* (denoted by p) and *opponent* (denoted by o) are introduced. The root of an attack–defense tree represents the main goal of the proponent. To be more precise, when the root is an attack node, the proponent is an attacker and the opponent is a defender, and vice versa.

To formalize attack–defense trees, we use attack–defense terms (ADTerms), which are either of the proponent's type, or of the opponent's type. ADTerms are defined in terms of basic actions of the proponent and opponent. Their basic actions are denoted with $\mathbb{B}^{\mathrm{p}}$ and $\mathbb{B}^{\mathrm{o}}$, respectively. The functions $\vee^{\mathrm{p}}, \wedge^{\mathrm{p}}, \vee^{\mathrm{o}}, \wedge^{\mathrm{o}}$ represent disjunctive ($\vee$) and conjunctive ($\wedge$) refinement operators of the proponent's and opponent's type, respectively. The binary function c^a ('*counter*'), where $a \in \mathbb{A}\mathrm{gt}$, connects a term of one agent's type with a countermeasure of the other agent's type.

The set of ADTerms T_Σ is defined as the set of all terms of the proponent's type.

Definition 8.1 (ADTerms)**.** *Given a set of basic actions of the proponent $\mathbb{B}^{\mathrm{p}}$ and a set of basic actions of the opponent $\mathbb{B}^{\mathrm{o}}$, we define the set of ADTterms of the proponent's type, denoted T^{p}_Σ, and the set of ADTerms of the opponent's type, denoted T^{o}_Σ, as follows.*

- $T^{\mathrm{p}}_\Sigma ::= \vee^{\mathrm{p}}(T^{\mathrm{p}}_\Sigma, \ldots, T^{\mathrm{p}}_\Sigma) \mid \wedge^{\mathrm{p}}(T^{\mathrm{p}}_\Sigma, \ldots, T^{\mathrm{p}}_\Sigma) \mid \mathrm{c}^{\mathrm{p}}(T^{\mathrm{p}}_\Sigma, T^{\mathrm{o}}_\Sigma) \mid \mathbb{B}^{\mathrm{p}}$;
- $T^{\mathrm{o}}_\Sigma ::= \vee^{\mathrm{o}}(T^{\mathrm{o}}_\Sigma, \ldots, T^{\mathrm{o}}_\Sigma) \mid \wedge^{\mathrm{o}}(T^{\mathrm{o}}_\Sigma, \ldots, T^{\mathrm{o}}_\Sigma) \mid \mathrm{c}^{\mathrm{o}}(T^{\mathrm{o}}_\Sigma, T^{\mathrm{p}}_\Sigma) \mid \mathbb{B}^{\mathrm{o}}$.

We define the set of ADTerms T_Σ as $T^{\mathrm{p}}_\Sigma \cup T^{\mathrm{o}}_\Sigma$. The type of an ADTerm, written $\tau(t)$, is defined by the function $\tau(t) = a$ where $t \in T^a_\Sigma$.

Example 8.1. *Consider the following ADTerm:*

$$t = \mathrm{c}^{\mathrm{p}}(\wedge^{\mathrm{p}}(D, E), \vee^{\mathrm{o}}(F)).$$

This ADTerm is graphically displayed in Figure 8.1a. In order to present the transformations between ADTerms and games more clearly, we draw counter notes slightly different from [KMRS10]. Our graphical depiction of ADTerms is basically a parse tree of such terms.

The depicted ADTerm represents an attacker that tries to gain access to a system by obtaining credentials to authenticate (A). The defender can prevent this attack by using two-factor authentication (C). The defender can implement two-factor authentication with key fobs (F). The attacker can log in with credentials if he obtains both the user name (D) and the password (E).

For this ADTerm, we have $\tau(t) = \mathrm{p}$. Subterms D and E are basic actions of the proponent's type, and F is a basic action of the opponent's type. Assuming that the proponent is the attacker, the system can be attacked by combining the basic attack actions D and E. However, the defender has the option to defend if he implements the basic defense action F.

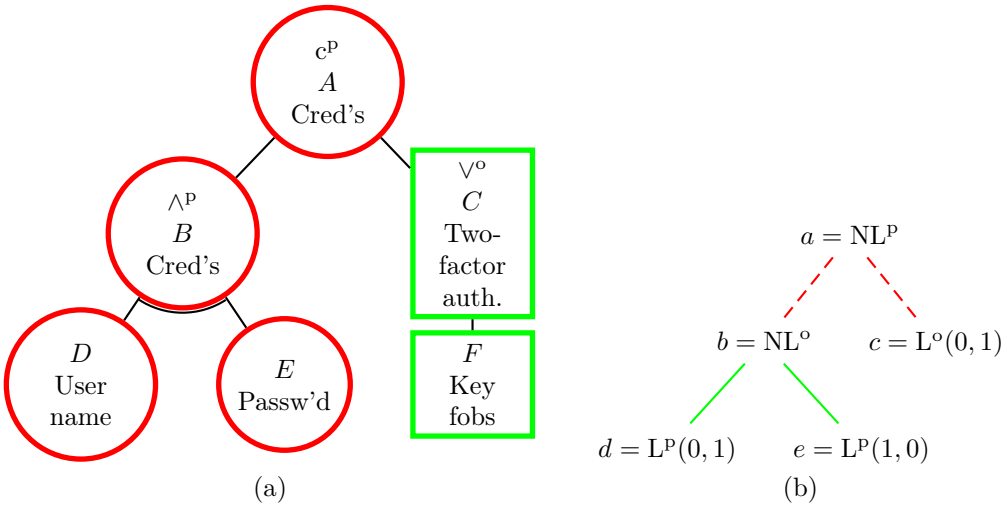


Figure 8.1: An example of an ADTerm (a) and a two-agent binary zero-sum extensive-form game (b).

In order to check whether an attack–defense scenario is feasible, the notion of satisfiability of an ADTerm is introduced by defining a satisfiability attribute sat_β. First, for agent $a \in \{\mathrm{p}, \mathrm{o}\}$, we define a *basic assignment* for a as a function $\beta^a \colon \mathbb{B}^a \to \{\mathsf{true}, \mathsf{false}\}$. We gather the basic assignments for both agents in a *basic assignment profile* $\beta = (\beta^{\mathrm{p}}, \beta^{\mathrm{o}})$. Second, given a basic assignment profile β, the function $sat_\beta \colon T_\Sigma \to \{\mathsf{true}, \mathsf{false}\}$ is used in order to calculate the satisfiability value of an ADTerm. It is defined recursively as follows.

$$sat_\beta(t) = \begin{cases} \beta^a(t), & \text{if } t \in \mathbb{B}^a, \\ \vee(sat_\beta(t_1), \ldots, sat_\beta(t_k)), & \text{if } t = \vee^a(t_1, \ldots, t_k), \\ \wedge(sat_\beta(t_1), \ldots, sat_\beta(t_k)), & \text{if } t = \wedge^a(t_1, \ldots, t_k), \\ sat_\beta(t_1) \wedge \neg sat_\beta(t_2), & \text{if } t = \mathrm{c}^a(t_1, t_2). \end{cases}$$

For instance, consider the term t from Example 8.1 and the basic assignment profile $\beta = (\beta^{\mathrm{p}}, \beta^{\mathrm{o}})$, where $\beta^{\mathrm{p}}(D) = \mathsf{true}$, $\beta^{\mathrm{p}}(E) = \mathsf{true}$, $\beta^{\mathrm{o}}(F) = \mathsf{false}$. We get $sat_\beta(t) = \mathsf{true}$. Assuming that the proponent is the attacker, the basic defense action F is absent and the system is attacked by combining the basic attack actions D and E.

The next definition formalizes the notion of a satisfiable ADTerm for an agent.

Definition 8.2 ($\mathrm{Sat}^a_\beta, \mathrm{Sat}^a_{\beta^a}, \mathrm{Sat}^a$)**.** *For every agent a, basic assignment β^a, and basic assignment profile β, we define the sets of ADTerms $\mathrm{Sat}^a_\beta, \mathrm{Sat}^a_{\beta^a}, \mathrm{Sat}^a \subseteq T_\Sigma$ in the following way. Let $t \in T_\Sigma$.*

- *Agent a is successful in t under basic assignment profile β, written $t \in \mathrm{Sat}^a_\beta$, if either $\tau(t) = a$ and $sat_\beta(t) = \mathsf{true}$, or $\tau(t) \neq a$ and $sat_\beta(t) = \mathsf{false}$.*
- *Agent a is successful in t under basic assignment β^a, written $t \in \mathrm{Sat}^a_{\beta^a}$, if $t \in \mathrm{Sat}^a_{(\beta^{\mathrm{p}}, \beta^{\mathrm{o}})}$ for every basic assignment β^{-a}.*
- *ADTerm t is satisfiable for a, written $t \in \mathrm{Sat}^a$, if there exists a basic assignment β^a for agent a such that $t \in \mathrm{Sat}^a_{\beta^a}$.*

The following theorem states that given an ADTerm t and a basic assignment profile β, either o or p is successful.

Theorem 8.1. *For every ADTerm t and basic assignment profile β, we have $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta}$ if and only if $t \notin \mathrm{Sat}^{\mathrm{o}}_{\beta}$.*

Proof. It holds that $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta}$ if and only if either $\tau(t) = \mathrm{p}$ and $sat_{\beta}(t) = \mathsf{true}$, or $\tau(t) = \mathrm{o}$ and $sat_{\beta}(t) = \mathsf{false}$. This is true exactly when neither $\tau(t) = \mathrm{p}$ and $sat_{\beta}(t) = \mathsf{false}$, nor $\tau(t) = \mathrm{o}$ and $sat_{\beta}(t) = \mathsf{true}$, which is equivalent to $t \notin \mathrm{Sat}^{\mathrm{o}}_{\beta}$. □

8.2.2 Two-agent Binary Zero-sum Extensive-Form Games

We consider *two-agent binary zero-sum extensive-form games*, in which a proponent p and an opponent o play against each other. In those games, we allow only for the outcomes $(1, 0)$ and $(0, 1)$, where $(1, 0)$ means that the proponent succeeds in his goal (breaking the system if he is the attacker, keeping the system secure if he is the defender), and $(0, 1)$ means that the opponent succeeds. Note that the proponent is not necessarily the agent who plays first in the game. Finally, we restrict ourselves to extensive-form games, i.e., games in tree format. We assume that turns in the game alternate between the two agents. To ease the translation of games to ADTerms, instead of using the usual notation, we present extensive-form games as terms. This gives us the following definition of two-agent binary zero-sum extensive-form games, where L stands for a leaf and NL for a non-leaf of the term.

Definition 8.3 (Two-agent binary zero-sum extensive-form game)**.** *Let $\mathbb{A}\mathrm{gt} = \{\mathrm{p}, \mathrm{o}\}$ denote the set of agents and let the set of outcomes be $\{(1, 0), (0, 1)\}$ the set of possible outcomes. A two-agent binary zero-sum extensive-form game is a term $t ::= \psi^{\mathrm{p}} \mid \psi^{\mathrm{o}}$, where*

$$\psi^{\mathrm{p}} ::= \mathrm{NL}^{\mathrm{p}}(\psi^{\mathrm{o}}, \ldots, \psi^{\mathrm{o}}) \mid \mathrm{L}^{\mathrm{p}}(1, 0) \mid \mathrm{L}^{\mathrm{p}}(0, 1)$$
$$\psi^{\mathrm{o}} ::= \mathrm{NL}^{\mathrm{o}}(\psi^{\mathrm{p}}, \ldots, \psi^{\mathrm{p}}) \mid \mathrm{L}^{\mathrm{o}}(1, 0) \mid \mathrm{L}^{\mathrm{o}}(0, 1).$$

We denote the set of all two-agent binary zero-sum extensive-form games by $\mathcal{G}^{*}_{\mathrm{E}}$.

Example 8.2. *An example of a two-agent binary zero-sum extensive-form game is the expression $\mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}(\mathrm{L}^{\mathrm{p}}(0, 1), \mathrm{L}^{\mathrm{p}}(1, 0)), \mathrm{L}^{\mathrm{o}}(0, 1))$. This game is displayed in Figure 8.1b. When displaying extensive-form games, we use dashed edges for choices made by the proponent, and solid edges for those made by the opponent. In this game, first the proponent can pick from two options; if he chooses the first option, the opponent can choose between outcomes $(0, 1)$ and $(1, 0)$. If the proponent chooses the second option, the game will end with outcome $(0, 1)$.*

Definition 8.4 (Extensive-form game terms)**.** *The set of extensive-form game terms consists of terms $\mathrm{NL}^{a}(\psi^{a'_1}, \ldots, \psi^{a'_m})$ and $\mathrm{L}^{a}(r^{a_1}, \ldots, r^{a_n})$, where $m, n \in \mathbb{R}$, $a, a'_1, \ldots, a'_m \in \mathbb{A}\mathrm{gt}$, $\psi^{a}, \ldots, \psi^{a}$ are extensive-form game terms, and $(r^{a_1}, \ldots, r^{a_n}) \in \mathbb{R}^{n}$.*

Every extensive-form game term belongs to an agent.

Definition 8.5 (Game term belonging to agent)**.** *If G is a game terms such that $G = \mathrm{NL}^{a}(\psi^{a'_1}, \ldots, \psi^{a'_m})$ or $G = \mathrm{L}^{a}(r^{a_1}, \ldots, r^{a_n})$, then we say that G belongs to agent a, written $\tau(G) = a$. We also say that ψ^{a} is a term of agent a.*

A *strategy* for a game for agent a is a function that assigns to every non-leaf one of the children of that leaf. A *strategy profile* is a combination of strategies, one for each agent.

Definition 8.6 (Strategy, strategy profile)**.** *A function σ^a is a strategy for game G for agent a if it assigns to every non-leaf $\mathrm{NL}^a(\psi_1^{a'}, \ldots, \psi_n^{a'})$ of agent a in game G a term $\psi_k^{a'}$ for some $k \in \{1, \ldots, n\}$.*

A strategy profile for a game G is a tuple $\sigma = (\sigma^{a_1}, \ldots, \sigma^{a_n})$, where σ^{a_i} is a strategy of agent a_i for game G.

For conciseness of notation, if $g = \mathrm{NL}^a(\psi_1^{a'}, \ldots, \psi_n^{a'})$ and $\sigma = (\sigma^{a_1}, \ldots, \sigma^{a_n})$, sometimes we write $\sigma(G)$ instead of $\sigma^a(G)$.

Definition 8.7 (Comparison of outcomes)**.** *We write*

$$(r_1^{a_1}, \ldots, r_1^{a_n}) \leq^{a_i} (r_2^{a_1}, \ldots, r_2^{a_n})$$

whenever $r_1^{a_i} \leq r_2^{a_i}$.

This definition implies that $(\mathbb{R}^n, \leq^a)$ is a totally ordered set for every $a \in \mathbb{A}\mathrm{gt}$.

We define the outcome under a strategy profile as the outcome we obtain when all agents play according to σ.

Definition 8.8 (Outcome under strategy profile)**.** *The* outcome out_σ *of a game G under strategy profile $\sigma = (\sigma^{a_1}, \ldots, \sigma^{a_n})$ is defined by:*

$$\begin{aligned} \mathrm{out}_\sigma(\mathrm{L}^a(r^{a_1}, \ldots, r^{a_n})) &= (r^{a_1}, \ldots, r^{a_n}) \\ \mathrm{out}_\sigma(\mathrm{NL}^a(\psi_1^{a'}, \ldots, \psi_n^{a'})) &= \mathrm{out}_\sigma(\sigma^a(\mathrm{NL}^a(\psi_1^{a'}, \ldots, \psi_n^{a'}))) \end{aligned}$$

We define the outcome under a strategy for a player as the outcome we obtain when the other player tries to maximize his own outcome.

Definition 8.9 (Outcome under strategy)**.** *Let $(r^{a_1}, \ldots, r^{a_n}) \in \mathbb{R}^n$, and $\psi_1^{a'}, \ldots, \psi_n^{a'}$ be games belonging to agent a'.*

The outcome out_{σ^a} *of a game g under strategy σ^a is defined by:*

$$\begin{aligned} \mathrm{out}_{\sigma^a}(\mathrm{L}^a(r^{a_1}, \ldots, r^{a_n})) &= (r^{a_1}, \ldots, r^{a_n}) \\ \mathrm{out}_{\sigma^a}(\mathrm{NL}^a(\psi_1^{a'}, \ldots, \psi_n^{a'})) &= \mathrm{out}_{\sigma^a}(\sigma^a(\mathrm{NL}^a(\psi_1^{a'}, \ldots, \psi_n^{a'}))) \\ \mathrm{out}_{\sigma^a}(\mathrm{NL}^{a'}(\psi_1^{a'}, \ldots, \psi_n^{a'})) &= \max_{1 \leq i \leq n}{}_{\leq^{a'}} \{\mathrm{out}_{\sigma^a}(\psi_i^{a'})\} \end{aligned}$$

Finally, we define the outcome of the game as the outcome we obtain when both agents try to maximize their own outcome.

Definition 8.10 (Outcome)**.** *Let $(r^{a_1}, \ldots, r^{a_n}) \in \mathbb{R}^n$, and let $\psi_1^{a'}, \ldots, \psi_n^{a'}$ be games belonging to agent a'.*

The outcome out *of a game g is defined by:*

$$\begin{aligned} \mathrm{out}(\mathrm{L}^a(r^{a_1}, \ldots, r^{a_n})) &= (r^{a_1}, \ldots, r^{a_n}) \\ \mathrm{out}(\mathrm{NL}^a(\psi_1^{a'}, \ldots, \psi_n^{a'})) &= \max_{1 \leq i \leq n}{}_{\leq^a} \{\mathrm{out}_{\sigma^a}(\psi_i^{a'})\} \\ \mathrm{out}(\mathrm{NL}^{a'}(\psi_1^{a'}, \ldots, \psi_n^{a'})) &= \max_{1 \leq i \leq n}{}_{\leq^{a'}} \{\mathrm{out}_{\sigma^a}(\psi_i^{a'})\} \end{aligned}$$

8.3 From Games to ADTerms

In this section, we show how to translate binary zero-sum two-agent extensive-form games to ADTerms. We do this by defining a function that translates games to equivalent ADTerms, and a function that translates strategies in games to basic assignments in the corresponding ADTerms. We show that these translations are correct in the following sense. First, we show that the agent who is winning in the game is also the agent for whom the corresponding ADTerm is satisfiable, if both agents play the basic assignment corresponding to their strategy in the game. Then we show that if an agent is winning in a game, he is successful in the corresponding ADTerm under the corresponding basic assignment. To translate games into ADTerms, we define a function $[\cdot]_{\mathrm{AD}}$.

Definition 8.11 (Translation of games to ADTerms)**.** *Let u, $u_1, \ldots, u_n$ represent fresh basic actions from $\mathbb{B}^{\mathrm{o}}$, and let v, $v_1, \ldots, v_n$ represent fresh basic actions from $\mathbb{B}^{\mathrm{p}}$. The function $[\cdot]_{\mathrm{AD}} \colon \mathcal{G}_{\mathrm{E}}^* \to T_\Sigma$ is defined in the following way.*

$$\mathrm{L}^{\mathrm{p}}(1,0) \mapsto v \tag{8.1a}$$
$$\mathrm{L}^{\mathrm{o}}(1,0) \mapsto \mathrm{c}^{\mathrm{o}}(u, v) \tag{8.1b}$$
$$\mathrm{L}^{\mathrm{p}}(0,1) \mapsto \mathrm{c}^{\mathrm{p}}(v, u) \tag{8.1c}$$
$$\mathrm{L}^{\mathrm{o}}(0,1) \mapsto u \tag{8.1d}$$
$$\mathrm{NL}^{\mathrm{p}}(\psi_1, \ldots, \psi_n) \mapsto \vee^{\mathrm{p}}(\mathrm{c}^{\mathrm{p}}(v_1, [\psi_1]_{\mathrm{AD}}), \ldots, \mathrm{c}^{\mathrm{p}}(v_n, [\psi_n]_{\mathrm{AD}})) \tag{8.1e}$$
$$\mathrm{NL}^{\mathrm{o}}(\psi_1, \ldots, \psi_n) \mapsto \vee^{\mathrm{o}}(\mathrm{c}^{\mathrm{o}}(u_1, [\psi_1]_{\mathrm{AD}}), \ldots, \mathrm{c}^{\mathrm{o}}(u_n, [\psi_n]_{\mathrm{AD}})) \tag{8.1f}$$

These rules are visualized in Figure 8.2. Rules (8.1a)–(8.1d) specify that a winning leaf for an agent in the game is translated to a satisfiable ADTerm for this agent, i.e., an ADTerm consisting of only a leaf belonging to this agent. Rule (8.1e)–(8.1f) specify that non-leaves in the game are translated to disjunctive ADTerms of the same agent. These disjunctions have children of the form $\mathrm{c}^a(u_k, [\psi_k]_{\mathrm{AD}})$ for some k. The intended meaning here is that agent a selects u_k exactly when his strategy selects ψ_k in the game.

Example 8.3. *Figure 8.3 depicts the translation of the game from Figure 8.1b to an ADTerm.*

The ADTerm resulting from $[\cdot]_{\mathrm{AD}} \colon \mathcal{G}_{\mathrm{E}}^* \to T_\Sigma$ is conjunction-free. Note that since terms in our games alternate between p and o, this translation results in correctly typed ADTerms.

Now we define how to translate a strategy profile for a game to a basic assignment profile for an ADTerm. First, we define a translation $[\![\cdot]\!]_{\mathrm{AD}}$ from a strategy σ^a ($a \in \{\mathrm{p}, \mathrm{o}\}$) for game G to a basic assignment $\beta^a = [\![\sigma^a]\!]_{\mathrm{AD}}^G$ for ADTerm $[G]_{\mathrm{AD}}$. Intuitively, if an agent's strategy for the game selects a certain branch, the basic assignment for the ADTerm assigns true to the node u_k in the corresponding branch, and false to the nodes u_k in the other branches. Furthermore, ADTerms resulting from leaves in the game are always assigned true by the basic assignment.

Definition 8.12 (Translation from strategy to basic assignment)**.** *Given a strategy σ^a for a in game G, we define $\beta^a = [\![\sigma^a]\!]_{\mathrm{AD}}^G$ as follows. For all ADTerms $\mathrm{c}^a(u, v)$*

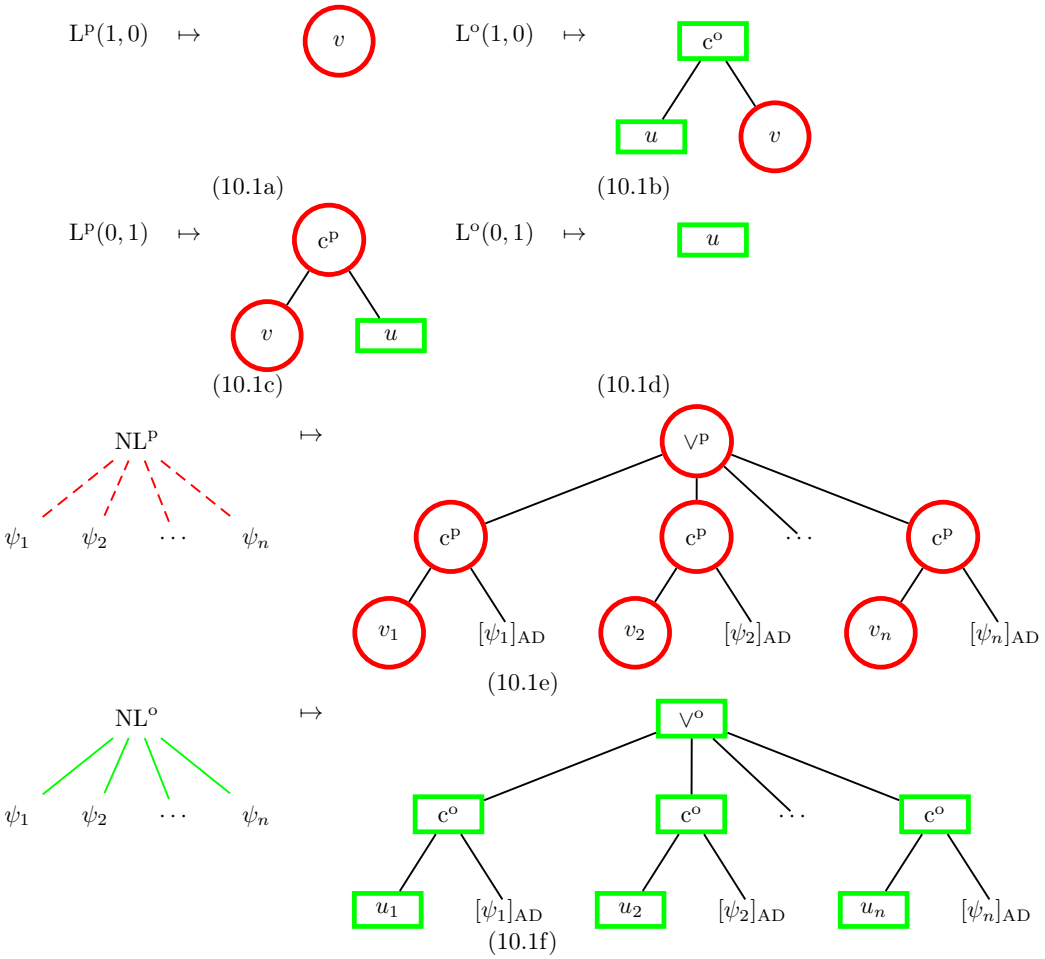


Figure 8.2: Translation of a game in extensive-form to an ADTerm by function $[\cdot]_{AD}$.

and v resulting from the first four cases in Definition 8.11, we set $\beta^a(u) = \beta^a(v) =$ true. For ADTerms obtained from game G by one of the last two cases in Definition 8.11, if $\sigma^a(G) = \psi_k$, we set $\beta^a(u_k) = \mathsf{true}$ and $\beta^a(u_i) = \mathsf{false}$ for $1 \leq i \leq n$, $i \neq k$.

We extend the translation $[\![\cdot]\!]_{AD}$ to strategy profiles as follows.

Definition 8.13 (Translation from strategy profile to basic assignment profile)**.** *We write $[\![(\sigma^p, \sigma^o)]\!]^G_{AD}$ for $([\![\sigma^p]\!]^G_{AD}, [\![\sigma^o]\!]^G_{AD})$.*

The next theorem states that an agent is winning in a game under a certain strategy profile if and only if he is successful in the corresponding ADTerm under the basic assignment profile corresponding to the strategy profile.

Theorem 8.2. *Let G be a game and let σ be a strategy profile for G. Then $\mathrm{out}_\sigma(G) = (1,0)$ if and only if $[G]_{AD} \in \mathrm{Sat}^p_{[\![\sigma]\!]^G_{AD}}$.*

Proof. Let G be a game and let σ a strategy profile for G. We set $t = [G]_{AD}$ and $\beta = [\![\sigma]\!]^G_{AD}$. To prove that $\mathrm{out}_\sigma(G) = (1,0)$ implies that $t \in \mathrm{Sat}^p_\beta$, we assume $\mathrm{out}_\sigma(G) = (1,0)$ and apply structural induction on G.

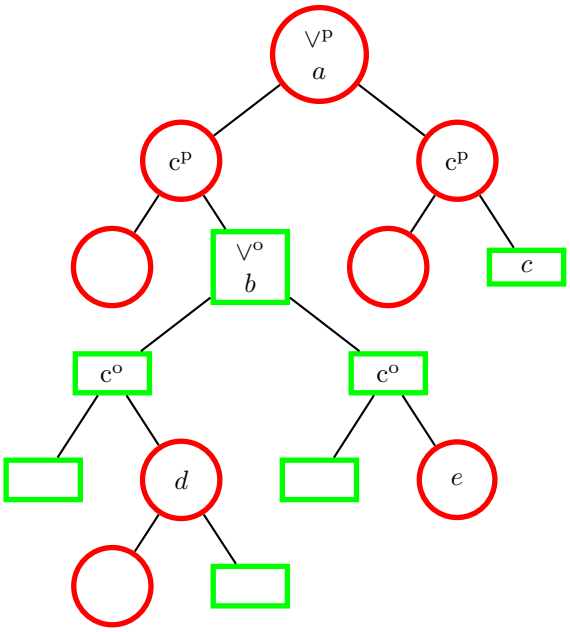


Figure 8.3: The result of the translation of the game from Figure 8.1b to an ADTerm.

In the base case, $G = \mathrm{L}^a(r^{\mathrm{p}}, r^{\mathrm{o}})$ with $r^{\mathrm{p}}, r^{\mathrm{o}} \in \{0, 1\}$. If $\mathrm{out}_\sigma(G) = (1, 0)$, then $r^{\mathrm{p}} = 1$ and $r^{\mathrm{o}} = 0$, so either $t = v$ or $t = \mathrm{c}^{\mathrm{o}}(u, v)$. In the first case, $sat_\beta(v) = \mathsf{true}$ by Definition 8.12, so $sat_\beta(t) = \mathsf{true}$, and therefore $t \in \mathrm{Sat}^{\mathrm{p}}_\beta$. In the second case, we have $sat_\beta(u) = sat_\beta(v) = \mathsf{true}$ by Definition 8.12, so $sat_\beta(t) = sat_\beta(\mathrm{c}^{\mathrm{o}}(u, v)) = \mathsf{false}$. Therefore, $t \notin \mathrm{Sat}^{\mathrm{o}}_\beta$, and thus $t \in \mathrm{Sat}^{\mathrm{p}}_\beta$.

To prove the induction step, we assume $G = \mathrm{NL}^a(\psi_1, \ldots, \psi_n)$. Then we have $t = \vee^a(\mathrm{c}^a(u_1, [\psi_1]_{\mathrm{AD}})), \ldots, \mathrm{c}^a(u_n, [\psi_n]_{\mathrm{AD}})$. Furthermore, there exists a k such that $\psi_k = \sigma(G)$. Therefore, $\mathrm{out}_\sigma(\psi_k) = (1, 0)$. By induction hypothesis, $[\psi_k]_{\mathrm{AD}} \in \mathrm{Sat}^{\mathrm{p}}_\beta$. First assume $a = \mathrm{p}$, so $\tau(\psi_k) = \mathrm{o}$ by the definition of a game. Then $\tau([\psi_k]_{\mathrm{AD}}) = \mathrm{o}$ by definition of $[\cdot]_{\mathrm{AD}}$, so $sat_\beta([\psi_k]_{\mathrm{AD}}) = \mathsf{false}$ by Definition 8.2 and $sat_\beta(u_k) = \mathsf{true}$ by definition of G. Therefore, $sat_\beta(\mathrm{c}^{\mathrm{p}}(u_k, [\psi_k]_{\mathrm{AD}})) = \mathsf{true}$, which implies $sat_\beta(t) = sat_\beta(\vee^a(\mathrm{c}^a(u_1, [\psi_1]_{\mathrm{AD}}), \ldots, \mathrm{c}^a(u_n, [\psi_n]_{\mathrm{AD}})) = \mathsf{true}$. Because it holds that $\tau(t) = \tau(G) = \mathrm{p}$, we have $t \in \mathrm{Sat}^{\mathrm{p}}_\beta$. Now assume $a = \mathrm{o}$, so $\tau(\psi_k) = \mathrm{p}$ by the definition of a game. Then $\tau([\psi_k]_{\mathrm{AD}}) = \mathrm{p}$ by definition of $[\cdot]_{\mathrm{AD}}$, so $sat_\beta([\psi_k]_{\mathrm{AD}}) = \mathsf{true}$ by Definition 8.2. Therefore, it holds that $sat_\beta(\mathrm{c}^{\mathrm{p}}(u_k, [\psi_k]_{\mathrm{AD}})) = \mathsf{false}$. Furthermore, whenever $1 \leq a \leq n$, $a \neq k$ $sat_\beta(u) = \mathsf{false}$ by definition of G, and therefore $sat_\beta(\mathrm{c}^{\mathrm{p}}(u, [\psi_k]_{\mathrm{AD}})) = \mathsf{false}$. This means that $sat_\beta(t) = sat_\beta(\vee^{\mathrm{o}}(\mathrm{c}^{\mathrm{o}}(u_1, [\psi_1]_{\mathrm{AD}}), \ldots, \mathrm{c}^{\mathrm{o}}(u_n, [\psi_n]_{\mathrm{AD}}))) = \mathsf{false}$. Because $\tau(t) = \tau(G) = \mathrm{o}$, $t \notin \mathrm{Sat}^{\mathrm{o}}_\beta$, and therefore $t \in \mathrm{Sat}^{\mathrm{p}}_\beta$.

Now we prove that $t \in \mathrm{Sat}^{\mathrm{p}}_\beta$ implies that $\mathrm{out}_\sigma(G) = (1, 0)$. We prove this by contraposition and structural induction on G. We set $t = [G]_{\mathrm{AD}}$ and $\beta = [\![\sigma]\!]^G_{\mathrm{AD}}$. In the base case, we have $\mathrm{out}_\sigma(G) \neq (1, 0)$, and thus $\mathrm{out}_\sigma(G) = (0, 1)$. By symmetry, we have $t \in \mathrm{Sat}^{\mathrm{o}}_\beta$, and therefore $t \notin \mathrm{Sat}^{\mathrm{p}}_\beta$. We proceed with the induction step. We have that $\mathrm{out}_\sigma(G) = (0, 1)$ implies $t \in \mathrm{Sat}^{\mathrm{o}}_\beta$ by symmetry. Then we also have $\mathrm{out}_\sigma(G) \neq (0, 1)$ implies $t \notin \mathrm{Sat}^{\mathrm{p}}_\beta$ by Theorem 8.1. □

The following theorem states that an agent is winning under a strategy in a game

if and only if that agent is successful in the corresponding ADTerm under the corresponding basic assignment. This is not a consequence of Theorem 8.2, as there might exist a basic assignment β^a for the ADTerm, for which there exists no strategy σ^a such that $\beta^a = [\![\sigma^a]\!]^G_{\mathrm{AD}}$ (i.e, the function $[\![\cdot]\!]_{\mathrm{AD}}$ is not surjective). Therefore, it is not immediately clear that if an agent has a strategy σ^a that wins from the other agent independent of his strategy, an agent with a basic assignment $[\![\sigma^a]\!]^G_{\mathrm{AD}}$ wins from the other agent independent of his basic assignment.

Theorem 8.3. *Let G be a game and σ^{p} be a strategy for* p *for G. Then* $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) = (1,0)$ *if and only if* $[G]_{\mathrm{AD}} \in \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$.

Proof. Let G be a game and let σ^{p} be a strategy for p for G. We set $t = [G]_{\mathrm{AD}}$ and $\beta^{\mathrm{p}} = [\![\sigma]\!]^G_{\mathrm{AD}}$. To prove that $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) = (1,0)$ implies $t \in \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$, we assume that $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) = (1,0)$ and apply structural induction on G.

In the base case, $G = \mathrm{L}^a(r^{\mathrm{p}}, r^{\mathrm{o}})$ with $r^{\mathrm{p}}, r^{\mathrm{o}} \in \{0,1\}$. Let $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$. If $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) = (1,0)$, then $r^{\mathrm{p}} = 1$ and $r^{\mathrm{o}} = 0$, so either $t = v$ or $t = \mathrm{c}^{\mathrm{o}}(u,v)$. In the first case, $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(v) = \mathsf{true}$ by Definition 8.12, so $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$, and therefore $t \in \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$, and thus $t \in \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$. In the second case, $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(u) = sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(v) = \mathsf{true}$ by Definition 8.12, so $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\mathrm{c}^{\mathrm{o}}(u,v)) = \mathsf{false}$. Therefore, $t \in \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$, and thus $t \in \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$.

We proceed with the induction step, i.e., $G = \mathrm{NL}^a(\psi_1, \ldots, \psi_n)$. This implies that $t = \vee^a(\mathrm{c}^a(u_1, [\psi_1]_{\mathrm{AD}}), \ldots, \mathrm{c}^a(u_n, [\psi_n]_{\mathrm{AD}}))$. We apply case distinction on a. First, we assume that $a = \mathrm{p}$. Then there exists k such that $\psi_k = \sigma^{\mathrm{p}}(G)$ and $\mathrm{out}_{\sigma^{\mathrm{p}}}(\psi_k) = (1,0)$. By the induction hypothesis, we have $[\psi_k]_{\mathrm{AD}} \in \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$. This implies that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we have $[\psi_k]_{\mathrm{AD}} \in \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. Therefore, since $\tau([\psi_k]_{\mathrm{AD}}) = \mathrm{o}$, we have for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}([\psi_k]_{\mathrm{AD}}) = \mathsf{false}$. Moreover, we have $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(u_k) = \mathsf{true}$ by definition of $[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}$. This implies that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we have that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\mathrm{c}^{\mathrm{p}}(u_k, [\psi_k]_{\mathrm{AD}})) = \mathsf{true}$, which in turn implies that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, it holds that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$. Since $\tau(G) = \mathrm{p}$, we have that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, it holds that $t \in \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. This implies $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. Now we assume that $a = \mathrm{o}$. We assume that $1 \leq k \leq n$. Then $\mathrm{out}_{\sigma^{\mathrm{p}}}(\psi_k) = (1,0)$, so $[\psi_k]_{\mathrm{AD}} \in \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$ by the induction hypothesis. This implies that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we have $[\psi_k]_{\mathrm{AD}} \in \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. Therefore, since $\tau([\psi_k]_{\mathrm{AD}}) = \mathrm{p}$, we have for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}([\psi_k]_{\mathrm{AD}}) = \mathsf{true}$. This implies that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we have $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\mathrm{c}^{\mathrm{o}}(u_k, [\psi_k]_{\mathrm{AD}})) = \mathsf{false}$, which in turn implies that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, it holds that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}([\psi_k]_{\mathrm{AD}}) = \mathsf{false}$. Since $\tau(t) = \mathrm{o}$, we have that for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, it holds that $t \in \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. This implies $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$.

To prove that $t \in \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$ implies $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) = (1,0)$, we assume that $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) \neq (1,0)$, i.e. $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) = (0,1)$, and prove that $t \neq \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$ by structural induction on G.

In the base case, $G = \mathrm{L}^a(r^{\mathrm{p}}, r^{\mathrm{o}})$ with $r^{\mathrm{p}}, r^{\mathrm{o}} \in \{0,1\}$. Let $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$. If $\mathrm{out}_{\sigma^{\mathrm{p}}}(G) = (1,0)$, then $r^{\mathrm{p}} = 1$ and $r^{\mathrm{o}} = 0$, so either $t = v$ or $t = \mathrm{c}^{\mathrm{o}}(u,v)$. In the first case, $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(v) = \mathsf{true}$ by Definition 8.12, so $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$, and therefore $t \notin \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$, and thus $t \notin \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$. In the second case, $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(u) = sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(v) = \mathsf{true}$ by Definition 8.12, so $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\mathrm{c}^{\mathrm{o}}(u,v)) = \mathsf{false}$.

Therefore, $t \notin \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$, and thus $t \notin \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$.

We proceed with the induction step, i.e., $G = \mathrm{NL}^a(\psi_1, \ldots, \psi_n)$. This implies that $t = \vee^a(\mathrm{c}^a(u_1, [\psi_1]_{\mathrm{AD}}), \ldots, \mathrm{c}^a(u_n, [\psi_n]_{\mathrm{AD}}))$. We apply case distinction on a. First, we assume that $a = \mathrm{p}$. Then there exists k such that $\psi_k = \sigma^{\mathrm{p}}(G)$ and $\mathrm{out}_{\sigma^{\mathrm{p}}}(\psi_k) = (0,1)$. This implies that $[\psi_k]_{\mathrm{AD}} \notin \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$ by the induction hypothesis. This implies that there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $[\psi_k]_{\mathrm{AD}} \notin \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. Therefore, since $\tau([\psi_k]_{\mathrm{AD}}) = \mathrm{o}$, there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}([\psi_k]_{\mathrm{AD}}) = \mathsf{true}$. This implies that there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\mathrm{c}^{\mathrm{o}}(u_k, [\psi_k]_{\mathrm{AD}})) = \mathsf{false}$, which in turn implies that there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}([\psi_k]_{\mathrm{AD}}) = \mathsf{false}$. Since $\tau(t) = \mathrm{p}$, there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $t \in \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. This implies $t \notin \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. Now we assume that $a = \mathrm{o}$. We assume that $1 \leq k \leq n$. Then $\mathrm{out}_{\sigma^{\mathrm{p}}}(\psi_k) = (0,1)$, so by the induction hypothesis, we have that $[\psi_k]_{\mathrm{AD}} \notin \mathrm{Sat}^{\mathrm{p}}_{[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}}$. This implies that there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $[\psi_k]_{\mathrm{AD}} \notin \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. Therefore, since $\tau([\psi_k]_{\mathrm{AD}}) = \mathrm{p}$, there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}([\psi_k]_{\mathrm{AD}}) = \mathsf{false}$. Moreover, we have $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(u_k) = \mathsf{true}$ by definition of $[\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}}$. This implies that there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\mathrm{c}^{\mathrm{p}}(u_k, [\psi_k]_{\mathrm{AD}})) = \mathsf{true}$, which in turn implies that there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$. Since $\tau(G) = \mathrm{o}$, there exists $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ such that $t \notin \mathrm{Sat}^{\mathrm{p}}_{([\![\sigma^{\mathrm{p}}]\!]^G_{\mathrm{AD}},\beta^{\mathrm{o}})}$. This implies $t \notin \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. □

Now we obtain the following corollary, which says that an agent is winning in a game if and only if it is successful in the corresponding ADTerm.

Corollary 8.1. *Whenever G is a game,* $\mathrm{out}(G) = (1,0)$ *if and only if* $[G]_{\mathrm{AD}} \in \mathrm{Sat}^{\mathrm{p}}$.

Proof. It holds that $\mathrm{out}(G) = (1,0)$ if and only if there is a strategy σ^{p} for p such that $\mathrm{out}_{\sigma^{\mathrm{p}}} = (1,0)$. This holds if and only if $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$ for some β^{p} by Theorem 8.3. This in turn is equivalent to $t \in \mathrm{Sat}^{\mathrm{p}}$. □

8.4 From ADTerms to Games

We proceed with the translation in the other direction, from ADTerms to games. We define two translations, one from ADTerms to games, and one from basic assignment profiles to strategy profiles. Then we show that if an agent has a basic assignment for an ADTerm with which he is successful, then that agent is winning in the corresponding game under he the corresponding strategy.

The translation from ADTerms to games is given by the following function $[\cdot]_{\mathrm{G}}$. A graphical representation of this translation is displayed in Figure 8.4.

It can easily be checked that this translation always generates valid games (in which p-moves and o-moves alternate).

Definition 8.14 (Translation from ADTerms to games)**.** *The function* $[\cdot]_{\mathrm{G}}$ *from*

ADTerms to games is defined as follows.

$$v \mapsto \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}(\mathrm{L}^{\mathrm{o}}(0,1),\mathrm{L}^{\mathrm{o}}(1,0))) \text{ if } v \in \mathbb{B}^{\mathrm{p}} \tag{8.2a}$$

$$v \mapsto \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}(\mathrm{L}^{\mathrm{p}}(1,0),\mathrm{L}^{\mathrm{p}}(0,1))) \text{ if } v \in \mathbb{B}^{\mathrm{o}} \tag{8.2b}$$

$$\vee^{\mathrm{p}}(\psi_1,\ldots,\psi_n) \mapsto \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}},\ldots,[\psi_n]_{\mathrm{G}})) \tag{8.2c}$$

$$\vee^{\mathrm{o}}(\psi_1,\ldots,\psi_n) \mapsto \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}},\ldots,[\psi_n]_{\mathrm{G}})) \tag{8.2d}$$

$$\wedge^{\mathrm{p}}(\psi_1,\ldots,\psi_n) \mapsto \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}}),\ldots,\mathrm{NL}^{\mathrm{p}}([\psi_n]_{\mathrm{G}})) \tag{8.2e}$$

$$\wedge^{\mathrm{o}}(\psi_1,\ldots,\psi_n) \mapsto \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}),\ldots,\mathrm{NL}^{\mathrm{o}}([\psi_n]_{\mathrm{G}})) \tag{8.2f}$$

$$\mathrm{c}^{\mathrm{p}}(\psi_1,\psi_2) \mapsto \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}}),[\psi_2]_{\mathrm{G}}) \tag{8.2g}$$

$$\mathrm{c}^{\mathrm{o}}(\psi_1,\psi_2) \mapsto \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}),[\psi_2]_{\mathrm{G}}) \tag{8.2h}$$

Rules (8.2a) and (8.2b) specify that leaves for agent a are translated to two options for agent a, a losing and a winning one. These choices correspond to not choosing and choosing the leaf in the ADTerm, respectively. Rules (8.2c) and (8.2d) specify that disjunctive terms for agent a are translated to choices for agent a in the game. There is no direct way of representing conjunctions in games. We can, however, still handle conjunctive terms, by translating them to choices for the other agent, as is specified by rules (8.2e) and (8.2f). This reflects the fact that an agent can succeed in all his options exactly when there is no way for the other agent to pick an option which allows him to succeed. Finally, rules (8.2g) and (8.2h) specify that countermeasures against agent a are translated to a choice for agent $-a$. Here, the first option corresponds to agent $-a$ not choosing the countermeasure, so that it is up to agent a whether he succeeds or not, while the second option corresponds to agent $-a$ choosing the countermeasure.

Example 8.4. *Figure 8.5 depicts the translation of the ADT from Figure 8.1a to a game.*

We proceed by defining a translation $[\![\cdot]\!]^{\cdot}_{\mathrm{G}}$ from a basic assignment for an ADTerm to a strategy for the corresponding game. We only give the definition for $a = \mathrm{p}$; the definition for $a = \mathrm{o}$ is symmetric.

Definition 8.15 (Translation from basic assignment to strategy)**.** *Function $[\![\cdot]\!]^{\cdot}_{\mathrm{G}}$ is a translation from a basic assignment β^{p} for ADTerm t to a strategy $\sigma^{\mathrm{p}} = [\![\beta^{\mathrm{p}}]\!]^{t}_{\mathrm{G}}$ for game $[t]_{\mathrm{G}}$. We define this translation using case distinction on t.*

- *Assume $t \in \mathbb{B}^{p}$. Then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}(\mathrm{L}^{\mathrm{o}}(0,1),\mathrm{L}^{\mathrm{o}}(1,0))$. We set $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = \mathrm{L}^{\mathrm{o}}(1,0)$ if $\beta^{a}(v) = \mathsf{true}$, and $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = \mathrm{L}^{\mathrm{o}}(1,0) = \mathrm{L}^{\mathrm{o}}(0,1)$ otherwise.*

- *Assume $t = \vee^{\mathrm{p}}(\psi_1,\ldots,\psi_n)$. Then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}},\ldots,[\psi_n]_{\mathrm{G}})$. We set $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = [\psi_k]_{\mathrm{G}}$ where k is the smallest number such that $\psi_k \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, and $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = [\psi_1]_{\mathrm{G}}$ if there exists no such number.*

- *Assume $t = \wedge^{\mathrm{p}}(\psi_1,\ldots,\psi_n)$. Then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}),\ldots,\mathrm{NL}^{\mathrm{o}}([\psi_n]_{\mathrm{G}}))$ We set $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = \mathrm{NL}^{\mathrm{o}}([\psi_k]_{\mathrm{G}})$ where k is the smallest number such that $\psi_k \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, and $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = \mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}})$ if there exists no such number.*

- *Assume $t = \mathrm{c}^{\mathrm{p}}(\psi_1,\psi_2)$, so $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}),[\psi_2]_{\mathrm{G}})$. We set $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = \mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}})$ if $\psi_2 \notin \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, and $\sigma^{\mathrm{p}}([t]_{\mathrm{G}}) = [\psi_2]_{\mathrm{G}}$ otherwise.*

Now we show that this translation is correct in the sense that agent p is successful in t under a basic assignment β^{p} if and only if p is winning with the strategy corresponding to this basic assignment in the corresponding game.

Theorem 8.4. *Let t be an ADTerm and let β^{p} be a basic assignment for t. Then $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$ if and only if $\mathrm{out}_{[\![\beta^{\mathrm{p}}]\!]^t_{\mathrm{G}}}([t]_{\mathrm{G}}) = (1,0)$.*

Proof. In order to prove that $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$ implies $\mathrm{out}_{[\![\beta^{\mathrm{p}}]\!]^t_{\mathrm{G}}}([t]_{\mathrm{G}}) = (1,0)$, we assume that $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. We prove that $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$ by structural induction on t. Since $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, we have that $\tau(t) = \mathrm{p}$ implies $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, and that $\tau(t) = \mathrm{o}$ implies $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$.

In the base case, we have either $t \in \mathbb{B}^{\mathrm{p}}$ or $t \in \mathbb{B}^{\mathrm{o}}$. If $t \in \mathbb{B}^{\mathrm{p}}$, we have that $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}(\mathrm{L}^{\mathrm{o}}(0,1), \mathrm{L}^{\mathrm{o}}(1,0)))$. Since $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we have $\beta^{\mathrm{p}}(t) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$ as well, so $\sigma^{\mathrm{p}}(\mathrm{NL}^{\mathrm{p}}(\mathrm{L}^{\mathrm{o}}(1,0), \mathrm{L}^{\mathrm{o}}(0,1))) = \mathrm{L}^{\mathrm{o}}(1,0)$ by Definition 8.15. Therefore, we have $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$. If $t \in \mathbb{B}^{\mathrm{o}}$, we have that $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}(\mathrm{L}^{\mathrm{p}}(1,0), \mathrm{L}^{\mathrm{p}}(0,1)))$. Then there exists a basic assignment β^{o} such that $t \notin \mathrm{Sat}^{\mathrm{p}}_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}$, namely when $\beta^{\mathrm{o}}(t) = \mathsf{true}$. Therefore, it holds that $t \notin \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, which is a contradiction.

We proceed with the induction step.

If $t = \vee^{\mathrm{p}}(\psi_1, \ldots, \psi_n)$, then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}}, \ldots, [\psi_n]_{\mathrm{G}}))$. Since $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, there exists j such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_j) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$. Let k be the smallest number with this property. Then $\psi_k \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, so $\sigma^{\mathrm{p}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}}, \ldots, [\psi_n]_{\mathrm{G}})) = [\psi_k]_{\mathrm{G}}$. Furthermore, since $\psi_k \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, we have $\mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_k]_{\mathrm{G}}) = (1,0)$ by the induction hypothesis. Therefore, $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$.

If $t = \vee^{\mathrm{o}}(\psi_1, \ldots, \psi_n)$, then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}, \ldots, [\psi_n]_{\mathrm{G}}))$. By $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, we have that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, so for $1 \leq k \leq n$, we have $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_k) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$. Furthermore, because $\tau(\psi_k) = \mathrm{o}$, it holds that $\psi_k \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. By the induction hypothesis, it holds that $\mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_k]_{\mathrm{G}}) = (1,0)$, so $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = \mathrm{out}_{\sigma^{\mathrm{p}}}(\mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}, \ldots, [\psi_n]_{\mathrm{G}}))) = (1,0)$.

If $t = \wedge^{\mathrm{p}}(\psi_1, \ldots, \psi_n)$, then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}}), \ldots, \mathrm{NL}^{\mathrm{p}}([\psi_n]_{\mathrm{G}}))$. Assume $1 \leq k \leq n$. By $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we obtain $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_k) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, and since $\tau(\psi_k) = \mathrm{p}$, we have $\psi_k \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. By the induction hypothesis, we have $\mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_k]_{\mathrm{G}}) = (1,0)$, so we have that $\mathrm{out}_{\sigma^{\mathrm{p}}}(\mathrm{NL}^{\mathrm{p}}([\psi_k]_{\mathrm{G}})) = (1,0)$ for $1 \leq k \leq n$, and therefore $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$.

If $t = \wedge^{\mathrm{o}}(\psi_1, \ldots, \psi_n)$, then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}), \ldots, \mathrm{NL}^{\mathrm{o}}([\psi_n]_{\mathrm{G}}))$. Because $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, there exists a number j such that $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_j) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$. Let k be the smallest number which this property. Then $\sigma^{\mathrm{p}}(t) = \mathrm{NL}^{\mathrm{o}}([\psi_k]_{\mathrm{G}})$. Furthermore, by $\psi_k \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, we have $\mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_k]_{\mathrm{G}}) = (1,0)$ by the induction hypothesis. Therefore $\mathrm{out}_{\sigma^{\mathrm{p}}}(w_k)) = (1,0)$ for $1 \leq k \leq n$, so $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$.

If $t = \mathrm{c}^{\mathrm{p}}(\psi_1, \psi_2)$, then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{o}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}}), [\psi_2]_{\mathrm{G}})$. By $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we obtain $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_1) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, and $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_2) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$. Since we have $\tau(\psi_1) = \mathrm{p}$ and $\tau(\psi_2) = \mathrm{o}$, both $\psi_1 \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$ and $\psi_2 \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. Then by the induction hypothesis, $\mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_1]_{\mathrm{G}}) = \mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_2]_{\mathrm{G}}) = (1,0)$. Then we have $\mathrm{out}_{\sigma^{\mathrm{p}}}(\mathrm{NL}^{\mathrm{p}}([\psi_1]_{\mathrm{G}})) = (1,0)$ as well, and therefore $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$.

If $t = \mathrm{c}^{\mathrm{o}}(\psi_1, \psi_2)$, then $[t]_{\mathrm{G}} = \mathrm{NL}^{\mathrm{p}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}}), [\psi_2]_{\mathrm{G}})$. Since we have $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(t) =$

false for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, we have either $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_1) = \mathsf{false}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$, or $sat_{(\beta^{\mathrm{p}},\beta^{\mathrm{o}})}(\psi_2) = \mathsf{true}$ for all $\beta^{\mathrm{o}} \in \mathbb{B}^{\mathrm{o}}$. In the first case, we have $\sigma^{\mathrm{p}}(t) = [\psi_1]_{\mathrm{G}}$. Moreover, by $\tau(\psi_1) = \mathrm{o}$, we have $\psi_1 \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, so $\mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_1]_{\mathrm{G}}) = (1,0)$ by the induction hypothesis. Then we have $\mathrm{out}_{\sigma^{\mathrm{p}}}(t) = (1,0)$ as well. In the second case, we have $\sigma^{\mathrm{p}}(t) = \mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}})$. Furthermore, by $\tau(\psi_2) = \mathrm{p}$, we have $\psi_2 \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$, which implies that $\mathrm{out}_{\sigma^{\mathrm{p}}}([\psi_2]_{\mathrm{G}}) = (1,0)$ by the induction hypothesis. Then $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = \mathrm{out}_{\sigma^{\mathrm{p}}}(\mathrm{NL}^{\mathrm{o}}([\psi_1]_{\mathrm{G}})) = (1,0)$.

Similarly, we can prove that $t \notin \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$ implies $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) \neq (1,0)$, which gives us by contraposition that $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$ implies $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$. □

Now the definition of out and $\mathrm{Sat}^{\mathrm{p}}$ gives us the following corollary. This gives us the following corollary, which states that an agent wins in a game if and only if the agent is successful in the corresponding ADTerm.

Corollary 8.2. *Whenever t is an ADTerm, $t \in \mathrm{Sat}^{\mathrm{p}}$ if and only if* $\mathrm{out}([t]_{\mathrm{G}}) = (1,0)$.

Proof. It holds that $t \in \mathrm{Sat}^{\mathrm{p}}$ if and only if $t \in \mathrm{Sat}^{\mathrm{p}}_{\beta^{\mathrm{p}}}$ for some β^{p}. By Theorem 8.4, this holds if and only if there exists a strategy σ^{p} for p such that $\mathrm{out}_{\sigma^{\mathrm{p}}}([t]_{\mathrm{G}}) = (1,0)$. This in turn is equivalent to $\mathrm{out}([t]_{\mathrm{G}}) = (1,0)$. □

8.5 Conclusion

We showed that attack–defense terms and binary zero-sum two-agent extensive-form games have equivalent expressive power when considering satisfiability, in the sense that they can be translated to each other while preserving their outcome. The translations preserve internal structure, in the sense that there exist injections between subterms in the game and subterms in the ADTerm such that if an agent wins in the subterm of the game, the corresponding subterm in the ADTerm is satisfiable for this agent, and vice versa. Therefore, attack–defense trees with a satisfiability attribute and binary zero-sum two-agent extensive-form games can be seen as two different representations of the same concept. Both representations have their own advantages. On the one hand, attack–defense trees are more intuitive, because conjunctions and refinements can be explicitly modeled. On the other hand, the game theory representation profits from the well-studied theoretical properties of games.

We saw that two notions in the domain of ADTerms have no direct correspondence to notions in the world of games: conjunctive nodes and refinements. Conjunctive nodes for one agent were translated to disjunctive nodes for the other agent. This also shows that, when considering the satisfiability attribute, the class of conjunction-free ADTerms has equal expressive power to the full class of ADTerms (note that the translation from ADTerms to games and vice versa are not each other's inverse, i.e., $[[G]_{\mathrm{AD}}]_{\mathrm{G}} \neq G$ and $[[t]_{\mathrm{G}}]_{\mathrm{AD}} \neq t$). Refinements were translated by adding extra dummy moves in between refining and refined nodes.

It is left for future work to translate attack–defense trees accompanied with more sophisticated attributes. An example of these are non-zero-sum games, where $(1,1)$ can be interpreted as an outcome where both the attacker and the defender profit

(for example, if the attacker pays the defender to achieve his goal), and $(0, 0)$ as an outcome where both parties are damaged (when the attacker fails in his goal, but his efforts damage the defender in some way). Also the binary requirement can be lifted, so that the outcome of an agent represents for instance the cost or gain of his actions. Furthermore, it would be interesting to look for a correspondence of incomplete and imperfect information in attack–defense trees.

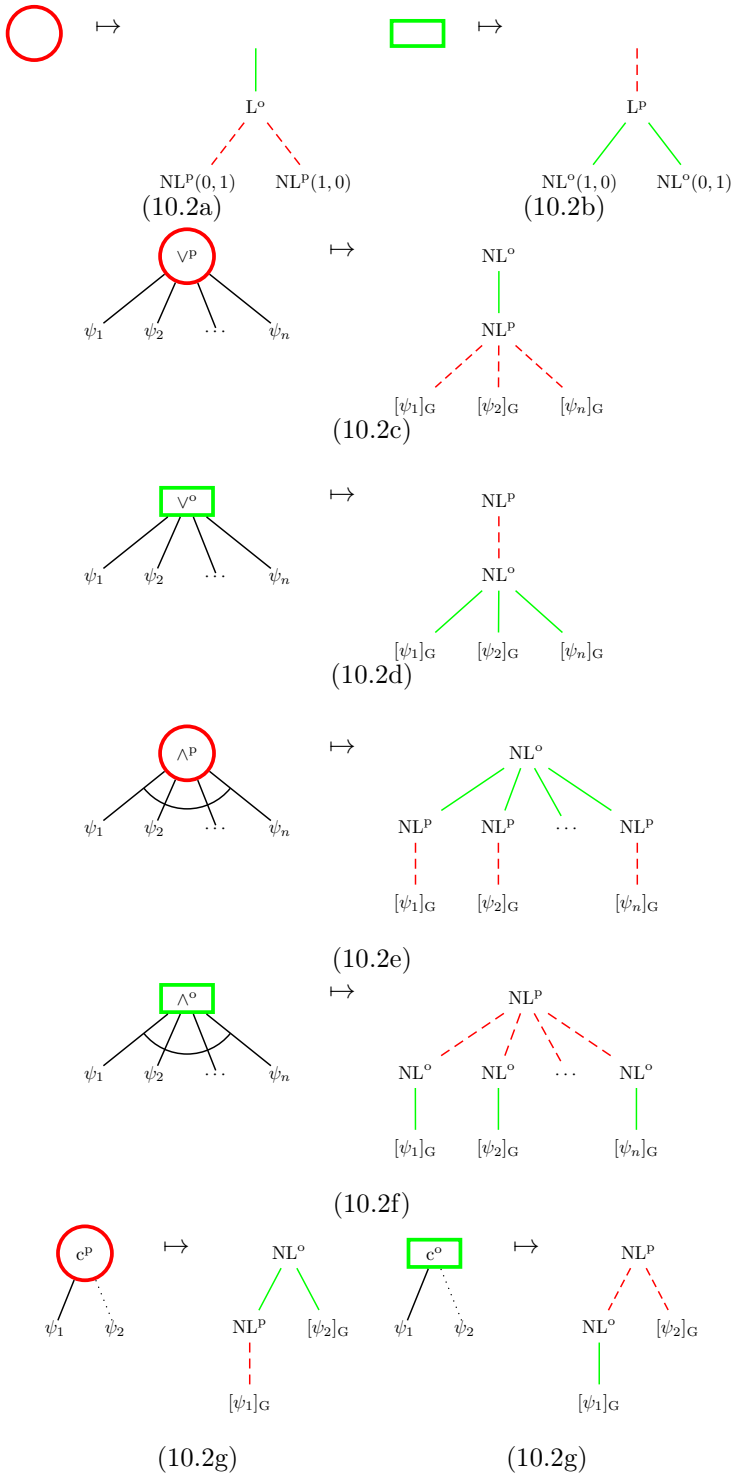


Figure 8.4: Translation of an ADTerm to a game by means of function $[\cdot]_G$.

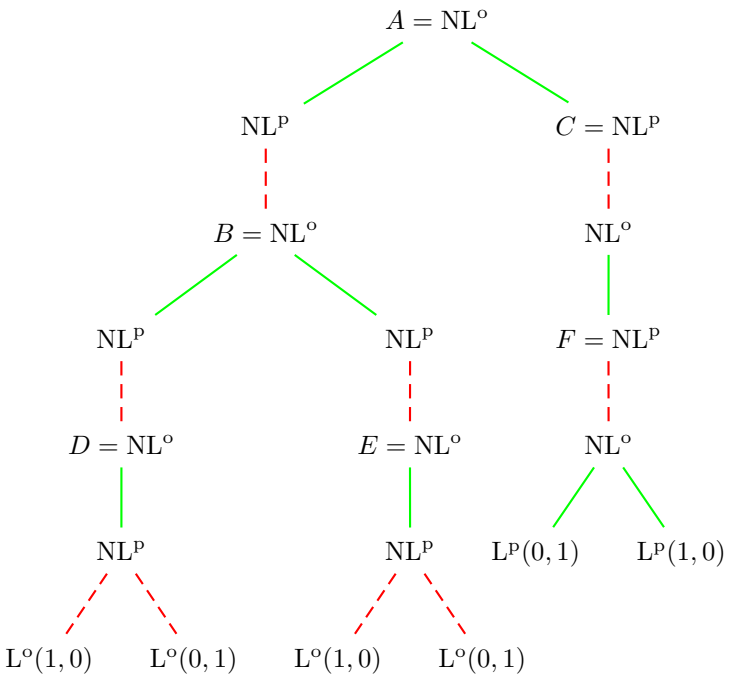


Figure 8.5: The result of the translation of the ADTerm from Figure 8.1a to a game.

9

Conclusion and Future Work

9.1 Conclusion

We restate the main research question of this thesis.

Research question: *How can game theory and logic help us to model the interaction between an attacker and a defender?*

To answer this question, we have divided the question in three subquestions. We first answer each of these subquestions individually.

Subquestion 1. How can game theory and logic help us to model the interaction between an attacker and a defender in security protocols?

We considered four aspects of this subquestion. First, we unified the Cremers–Mauw protocol semantics with concurrent game structures. This allowed us to define security properties on Cremers–Mauw protocol specifications in Alternating-time Temporal Logic.

Second, we studied imperfect information of participants in security protocols. This resulted in the discovery of two limitations of the protocol verification framework of Kremer and Raskin [KR03]. The first limitation has to do with the fact that perfect information is implicitly assumed in their framework. The second limitation is the fact that the properties of fairness and effectiveness cannot be combined easily. We proposed a solution for both limitations individually. Moreover, we established a hierarchy of various definitions of fairness, and indicated the consequences for existing work. To investigate whether it is possible to overcome both limitations at the same time, we compared the expressive power of ATL* and the simple fragment of Strategy Logic. We showed that on turn-based game structure, the unnested simple one-alternating fragment of Strategy Logic without $\bigcirc$-operator is at least as expressive as ATL*. We did so by providing a translation from the former fragment into ATL*. On concurrent game structures, however, we showed that the simple fragment of Strategy Logic is in fact more expressive than ATL*. This is also the case on imperfect-information game structures, even when they are turn-based. These results imply that ATL* is not expressive enough to overcome both limitations at the same time.

Third, we proposed a formal definition of the security property of non-repudiation, and we studied the knowledge assumptions that agents need to have about the behavior of the other agents. When these assumption are not satisfied, a new class of attacks arises, which we called virtual multi-protocol attacks. In addition, we introduced an additional type of non-repudiation, which we called non-repudiation of intention.

Forth, we proposed a methodological framework for analyzing interaction protocols that takes into account the incentives of agents. The framework allows for a more fine-grained analysis of such protocols. We formally defined correctness of a protocol given a notion of rationality, and possibly given the utilities of the agents and a set of agents that supports the objective of the protocol. Finally, we described the security level of a protocol as the minimal sets of participants supporting the objective of the protocol that are necessary to achieve this objective.

All aspects that were studied lead to new insights related to security protocols and their verification. We conclude therefore that logic and game theory are fruitful methodologies for the study of security protocols.

Subquestion 2. How can game theory and logic help us to model the interaction between an attacker and a defender in farsighted games?

We proposed a new solution concept, which we called *farsighted pre-equilibrium*. It is based on the idea that some locally rational deviations in a strategic-form game, which can be seen as a player 'attacking' the stable situation, may not be profitable anymore if one takes into account the possibility of further 'defenses' by the other players. We proved that positional strategies are sufficient to define the concept, studied its computational complexity, and showed that pre-equilibria correspond to subgame-perfect Nash equilibria in a meta-game obtained by using the original payoff matrix as arena and the deviations as moves.

Subquestion 3. How can game theory and logic help us to model the interaction between an attacker and a defender in the field of attack modeling?

We studied attack-defense trees, and made the connection between attack-defense trees and game theory explicit. To be more precise, we showed that attack-defense trees and binary zero-sum two-player extensive form games have equivalent expressive power when considering satisfiability, in the sense that it is possible to convert ADTerms to these games (and vice-versa) while preserving their outcome and their internal structure.

We conclude that in all three studied areas, the research conducted has shown that logic and game theory can lead to new insights in security. In all three areas, it turned out that we should not only look into attacks carried out by the attacker, but also consider the possibility for the defender to prevent the attacker. In fair-exchange protocols, an attacker might be able to get an item from the other player, while the other player does not yet have the item of the attacker. This is not a problem, however, as long as the other player has the ability to react and get the defender's item. In farsighted games, an agent might try to deviate from a strategy profile. However, this could be seen as an unsuccessful 'attack' when other agents can follow up with deviations that make the initial deviator be worse off. In attack modeling, only modeling the attacks is not sufficient either. In general, it turns out that we can only get a complete view of the vulnerabilities of a system by not only looking at the possible attacks, but also at possible countermeasures.

9.2 Future Work

We expect that our methodology can also be extended to investigate other aspects of security protocols. A first possible extension is to investigate applications of different kinds of transition systems in security. Transition systems, of which concurrent game structures are an extension, are often used as underlying model for verification. The application of imperfect-information models, which are one of these extensions, has been analyzed in this thesis. Another way to extend transition systems is to make use of probabilistic models. Current work in security focuses on fully guaranteeing security. However, in certain circumstances, a probability of security might be good enough, especially when the measures that guarantee security are costly. It might be sufficient that a system is protected almost surely, i.e., with probability one, instead of surely (see e.g. [BCW+09]), or even with a lower probability. It would be interesting to use such models for the verification of security properties. Probabilistic models that could be used include QAPI [Sch12] and pATL* [CL07]. Another recent extension of transition systems are models that explicitly refer to time. An example of such a model is Timed Alternating-time Temporal Logic, which is interpreted in Durational Concurrent Game Structures [HP06, LMO06]. It would also be interesting to see whether such models have a purpose in security.

This thesis has pointed out a number of theoretical problems, such as lack of information to play the right strategy (Chapter 3), and virtual multi-protocol attacks (Chapter 5). We looked at hypothetical protocols in which these problems occur. However, we did not find any real-life protocols that are vulnerable to such problems. Future research could focus on how these theoretical problems could be exploited in real life.

To enable such research, it would be helpful to be able to do automatic verification of security properties in imperfect-information models. There are currently some obstacles that prevent this. First, it is known that ATL* model-checking in models with imperfect information and perfect recall (i.e., memory-based strategies) is undecidable [DT11]. Even with imperfect recall, the model checking problem is in PSPACE. Due to these facts, few tools are available for imperfect-information model checking. To do so, Alpaga [BCW+09], which is a tool for solving parity games with imperfect information, seems a good start. However, at its current state, Alpaga is not suitable for the verification of fair-exchange protocols, as Alpaga requires observable objectives, while objectives in fair-exchange protocols are observable.

Another way to extend the results of this thesis is to consider other security properties. In this thesis, we focused on the security property of non-repudiation. We expect that our methodology can be extended to other security properties as well.

A first logical choice would be *abuse-freeness* [GM99, KST10]. A protocol is abuse-free if no signer S is able to prove to an external observer that S has the power to choose between successfully concluding the protocol and aborting the protocol. This property is interesting because it cannot be formulated as a property that should hold in all paths, and thus requires explicit reasoning about strategies to define it.

Furthermore, it would be interesting to define *accountability* [KTV10, KR10] in a way similar to non-repudiation. Accountability is typically modeled as two properties: first if an agent has evidence of an event, then the event has happened; second, if an event happens, then the agent will have evidence of this event. Superficially, these properties look similar to the reachability and soundness of non-repudiation, respectively (Chapter 5).

In Chapter 5, we have seen that authentication can be seen as a special case of non-repudiation. It would be also interesting to investigate whether there exist non-repudiation-like properties that correspond to other forms of authentication, such as injective agreement [DBL97].

Furthermore, there are some technical issues left to analyze. In this thesis, as well as in Cremers–Mauw protocol specifications, the attacker in a security protocol is implicit, as it is modeled by the injection, blocking and deleting of messages in the network. Alternatively, it is possible to model the attacker either as an agent on its own that is not restricted by any protocol, or by means of a separately crafted protocol (as is the case in multi-protocol attacks). It would be interesting to formally investigate whether these three ways of modeling the attacker yield equally strong attackers.

Finally, it would be interesting to see if it is possible to do *mechanism design* by manipulating available information of protocol participants. Mechanism design is closely related to game theory. While game theory predicts the behavior of agents when given a game as input, mechanism design aims to design a game with the purpose of making agents display desired behavior. Mechanism design is typically done by altering incentives of agents. However, it might also be possible to influence the behavior of agents by manipulating their initial knowledge. This leads to the question whether we can distribute information among agents in a security protocol in such a way as to satisfy the security property.

Bibliography

[AB06] Eugene Asarin and Patricia Bouyer, editors. *Formal Modeling and Analysis of Timed Systems, 4th International Conference, FORMATS 2006, Paris, France, September 25-27, 2006, Proceedings*, volume 4202 of *Lecture Notes in Computer Science*. Springer, 2006.

[ACH11] Gilad Asharov, Ran Canetti, and Carmit Hazay. Towards a game theoretic view of secure computation. In Kenneth G. Paterson, editor, *EUROCRYPT*, volume 6632 of *Lecture Notes in Computer Science*, pages 426–445. Springer, 2011.

[AE11] Alfin Abraham and Vinodh Ewards. An optimistic fair RSA based protocol for e-commerce. In *International Conference on VLSI, Communications and Instrumentation*, pages 16–19, 2011.

[ÅGJ07] Thomas Ågotnes, Valentin Goranko, and Wojciech Jamroga. Alternating-time temporal logics with irrevocable strategies. In Dov Samet, editor, *TARK*, pages 15–24, 2007.

[AHK02] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.

[AHM+98] Rajeev Alur, Thomas A. Henzinger, Freddy Y. C. Mang, Shaz Qadeer, Sriram K. Rajamani, and Serdar Tasiran. Mocha: Modularity in model checking. In Alan J. Hu and Moshe Y. Vardi, editors, *CAV*, volume 1427 of *Lecture Notes in Computer Science*, pages 521–525. Springer, 1998.

[Aiz08] Mihhail Aizatulin. A timely and balanced optimistic contract-signing protocol. Master's thesis, Institut für Informatik, Christian-Albrechts-Universität zu Kiel, 2008.

[AL11] Gilad Asharov and Yehuda Lindell. Utility dependence in correct and fair rational secret sharing. *J. Cryptology*, 24(1):157–202, 2011.

[Ame] Amenaza. SecurITree. `http://www.amenaza.com/`.

[AMNO07] Ross Anderson, Tyler Moore, Shishir Nagaraja, and Andy Ozment. Incentives and information security. In Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V. Vazirani, editors, *Algorithmic Game Theory*, pages 633–649. Cambridge University Press, New York, NY, USA, 2007.

[Aso98] N Asokan. *Fairness in Electronic Commerce*. PhD thesis, University of Waterloo, may 1998.

[ASW98] N. Asokan, Victor Shoup, and Michael Waidner. Asynchronous protocols for optimistic fair exchange. In *IEEE Symposium on Security and Privacy*, pages 86–99. IEEE Computer Society, 1998.

[ASW09] Mihhail Aizatulin, Henning Schnoor, and Thomas Wilke. Computationally sound analysis of a probabilistic contract signing protocol. In Backes and Ning [BN09], pages 571–586.

[BBP94] Jan A. Bergstra, Inge Bethke, and Alban Ponse. Process algebra with iteration and nesting. *Comput. J.*, 37(4):243–258, 1994.

[BCW+09] Dietmar Berwanger, Krishnendu Chatterjee, Martin De Wulf, Laurent Doyen, and Thomas A. Henzinger. Alpaga: A tool for solving parity games with imperfect information. In Stefan Kowalewski and Anna Philippou, editors, *TACAS*, volume 5505 of *Lecture Notes in Computer Science*, pages 58–61. Springer, 2009.

[BDP06] Stefano Bistarelli, Marco Dall'Aglio, and Pamela Peretti. Strategic games on defense trees. In Theodosis Dimitrakos, Fabio Martinelli, Peter Y. A. Ryan, and Steve A. Schneider, editors, *Formal Aspects in Security and Trust*, volume 4691 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2006.

[BHC04] Levente Buttyán, Jean-Pierre Hubaux, and Srdjan Capkun. A formal model of rational exchange and its application to the analysis of syverson's protocol. *Journal of Computer Security*, 12(3-4):551–587, 2004.

[BK08] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

[BN09] Michael Backes and Peng Ning, editors. *Computer Security - ESORICS 2009, 14th European Symposium on Research in Computer Security, Saint-Malo, France, September 21-23, 2009. Proceedings*, volume 5789 of *Lecture Notes in Computer Science*. Springer, 2009.

[BOGMR90] Michael Ben-Or, Oded Goldreich, Silvio Micali, and Ronald L. Rivest. A fair protocol for signing contracts. *IEEE Transactions on Information Theory*, 36(1):40–46, 1990.

[BP01] Giampaolo Bella and Lawrence C. Paulson. A proof of non-repudiation. In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, *Security Protocols Workshop*, volume 2467 of *Lecture Notes in Computer Science*, pages 119–125. Springer, 2001.

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures - how to sign with RSA and rabin. In Ueli M. Maurer, editor, *EUROCRYPT*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer, 1996.

[Bra87] Michael E. Bratman. *Intention, Plans, and Practical Reason*. Harvard University Press, November 1987.

[BT94] Josh Cohen Benaloh and Dwight Tuinstra. Receipt-free secret-ballot elections (extended abstract). In Frank Thomson Leighton and Michael T. Goodrich, editors, *STOC*, pages 544–553. ACM, 1994.

[CD06] Jan Cederquist and Muhammad Torabi Dashti. An intruder model for verifying liveness in security protocols. In Marianne Winslett, Andrew D. Gordon, and David Sands, editors, *FMSE*, pages 23–32. ACM, 2006.

[CHJ06] Krishnendu Chatterjee, Thomas A. Henzinger, and Marcin Jurdzinski. Games with secure equilibria. *Theor. Comput. Sci.*, 365(1-2):67–82, 2006.

[CHP10] Krishnendu Chatterjee, Thomas A. Henzinger, and Nir Piterman. Strategy logic. *Inf. Comput.*, 208(6):677–693, 2010.

[Chw94] Michael S. Chwe. Farsighted coalitional stability. *Journal of Economic Theory*, 63:299–325, 1994.

[CKS01] Rohit Chadha, Max I. Kanovich, and Andre Scedrov. Inductive methods and contract-signing protocols. In Michael K. Reiter and Pierangela Samarati, editors, *ACM Conference on Computer and Communications Security*, pages 176–185. ACM, 2001.

[CKS06] Rohit Chadha, Steve Kremer, and Andre Scedrov. Formal analysis of multiparty contract signing. *J. Autom. Reasoning*, 36(1-2):39–83, 2006.

[CL07] Taolue Chen and Jian Lu. Probabilistic alternating-time temporal logic and model checking algorithm. In J. Lei, editor, *FSKD (2)*, pages 35–39. IEEE Computer Society, 2007.

[CM03] Cas J. F. Cremers and Sjouke Mauw. Operational semantics of security protocols. In Stefan Leue and Tarja Systä, editors, *Scenarios: Models, Transformations and Tools*, volume 3466 of *Lecture Notes in Computer Science*, pages 66–89. Springer, 2003.

[CM12] Cas J.F. Cremers and Sjouke Mauw. *Operational Semantics and Verification of Security Protocols*. Information Security and Cryptography. Springer, 2012.

[CMdV06] Cas J. F. Cremers, Sjouke Mauw, and Erik P. de Vink. Injective synchronisation: An extension of the authentication hierarchy. *Theor. Comput. Sci.*, 367(1-2):139–161, 2006.

[CMSS05] Rohit Chadha, John C. Mitchell, Andre Scedrov, and Vitaly Shmatikov. Contract signing, optimism, and advantage. *J. Log. Algebr. Program.*, 64(2):189–218, 2005.

[CR10] Krishnendu Chatterjee and Vishwanath Raman. Assume-guarantee synthesis for digital contract signing. *CoRR*, abs/1004.2697, 2010.

[Cre06] Cas J. F. Cremers. Feasibility of multi-protocol attacks. In *ARES*, pages 287–294. IEEE Computer Society, 2006.

[CVE] CVE-2012-6578. `http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-6578`.

[Dav01] Don Davis. Defective sign & encrypt in S/MIME, PKCS#7, MOSS, PEM, PGP, and XML. In Yoonho Park, editor, *USENIX Annual Technical Conference, General Track*, pages 65–78. USENIX, 2001.

[DBL97] *10th Computer Security Foundations Workshop (CSFW '97), June 10-12, 1997, Rockport, Massachusetts, USA*. IEEE Computer Society, 1997.

[DJ10] Mehdi Dastani and Wojciech Jamroga. Reasoning about strategies of multi-agent programs. In van der Hoek et al. [vdHKL+10], pages 997–1004.

[DKR06] Stéphanie Delaune, Steve Kremer, and Mark Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *CSFW*, pages 28–42. IEEE Computer Society, 2006.

[DT11] Catalin Dima and Ferucio Laurentiu Tiplea. Model-checking ATL under imperfect information and perfect recall semantics is undecidable. *CoRR*, abs/1102.4225, 2011.

[Dun95] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[DX03] Effrosyni Diamantoudi and Licun Xue. Farsighted stability in hedonic games. *Social Choice and Welfare*, 21(1):39–61, 2003.

[EDRM06] Kenneth S. Edge, George C. Dalton, II, Richard A. Raines, and Robert F. Mills. Using attack and protection trees to analyze threats and defenses to homeland security. In *Proceedings of the 2006 IEEE conference on Military communications*, MILCOM'06, pages 953–959, Piscataway, NJ, USA, 2006. IEEE Press.

[EH85] E. Allen Emerson and Joseph Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *Journal of Computer and System Sciences*, 30(1):1–24, February 1985.

[ES03] Paul D. Ezhilchelvan and Santosh K. Shrivastava. Systematic development of a family of fair exchange protocols. In Sabrina De Capitani di Vimercati, Indrakshi Ray, and Indrajit Ray, editors, *DBSec*, pages 243–258. Kluwer, 2003.

[EY80] Shimon Even and Yacov Yacobi. Relations among public key signature systems. Technical Report 175, Technion, Haifa, Israel, March 1980.

[FHMV95] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning about Knowledge*. MIT Press, 1995.

[Fok96] Wan Fokkink. On the completeness of the equations for the Kleene star in bisimulation. In Martin Wirsing and Maurice Nivat, editors, *AMAST*, volume 1101 of *Lecture Notes in Computer Science*, pages 180–194. Springer, 1996.

[Fok97] Wan Fokkink. Axiomatizations for the perpetual loop in process algebra. In Pierpaolo Degano, Roberto Gorrieri, and Alberto Marchetti-Spaccamela, editors, *ICALP*, volume 1256 of *Lecture Notes in Computer Science*, pages 571–581. Springer, 1997.

[FS10] Bernd Finkbeiner and Sven Schewe. Coordination logic. In Anuj Dawar and Helmut Veith, editors, *CSL*, volume 6247 of *Lecture Notes in Computer Science*, pages 305–319. Springer, 2010.

[Gar95] Simson L. Garfinkel. *PGP - pretty good privacy: encryption for everyone (2. ed.)*. O'Reilly, 1995.

[GJM99] Juan A. Garay, Markus Jakobsson, and Philip D. MacKenzie. Abuse-free optimistic contract signing. In Michael J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 449–466. Springer, 1999.

[GK12] Adam Groce and Jonathan Katz. Fair computation with rational players. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 81–98. Springer, 2012.

[GLL07] Hojjat Ghaderi, Hector J. Levesque, and Yves Lespérance. A logical theory of coordination and joint ability. In *AAAI*, pages 421–426. AAAI Press, 2007.

[GM99] Juan A. Garay and Philip D. MacKenzie. Abuse-free multi-party contract signing. In Prasad Jayanti, editor, *DISC*, volume 1693 of *Lecture Notes in Computer Science*, pages 151–165. Springer, 1999.

[Gre90] Joseph Greenberg. *The theory of social situations: an alternative game-theoretic approach*. Cambridge University Press, 1990.

[GT00] Joshua D. Guttman and F. Javier Thayer. Protocol independence through disjoint encryption. In *CSFW*, pages 24–34, 2000.

[Gua13] The Guardian. The NSA is turning the internet into a total surveillance system. `http://www.theguardian.com/commentisfree/2013/aug/11/nsa-internet-surveillance-email`, 2013. Accessed: 2013-09-01.

[Har74] John C. Harsanyi. Interpretation of stable sets and a proposed alternative definition. *Management Science*, 20:1472–1495, 1974.

[HLS03] James Heather, Gavin Lowe, and Steve Schneider. How to prevent type flaw attacks on security protocols. *Journal of Computer Security*, 11(2):217–244, 2003.

[HP06] Thomas A. Henzinger and Vinayak S. Prabhu. Timed alternating-time temporal logic. In Asarin and Bouyer [AB06], pages 1–17.

[HR10] Joseph Y. Halpern and Nan Rong. Cooperative equilibrium. In van der Hoek et al. [vdHKL+10], pages 1465–1466.

[Iso] Isograph. AttackTree+. `http://www.isograph-software.com/atpover.htm`.

[JB11] Wojciech Jamroga and Nils Bulling. Comparing variants of strategic ability. In Toby Walsh, editor, *IJCAI*, pages 252–257. IJCAI/AAAI, 2011.

[JM11] Wojciech Jamroga and Matthijs Melissen. Doubtful deviations and farsighted play. In Luis Antunes and Helena Sofia Pinto, editors, *Progress in Artificial Intelligence, 15th Portuguese Conference on Artificial Intelligence, EPIA 2011, Lisbon, Portugal, October 10-13, 2011. Proceedings*, volume 7026 of *Lecture Notes in Computer Science*, pages 506–520. Springer, 2011.

[JMM12] Wojciech Jamroga, Sjouke Mauw, and Matthijs Melissen. Fairness in Non-Repudiation Protocols. In Catherine Meadows and M. Carmen Fernández Gago, editors, *Proceedings of the 7th International Workshop on Security and Trust Management*, volume 7170 of *Lecture Notes in Computer Science*, pages 122–139. Springer, June 2012.

[JMS13] Wojciech Jamroga, Matthijs Melissen, and Henning Schnoor. Incentives and rationality in security of interaction protocols. In *PRIMA 2013: Principles and Practice of Multi-Agent Systems. Proceedings*, 2013. In print.

[JvdH04] Wojciech Jamroga and Wiebe van der Hoek. Agents that know how to play. *Fundam. Inform.*, 63(2-3):185–219, 2004.

[JW09] Aivo Jürgenson and Jan Willemson. Serial model for attack tree computations. In Donghoon Lee and Seokhie Hong, editors, *ICISC*, volume 5984 of *Lecture Notes in Computer Science*, pages 118–128. Springer, 2009.

[Kal98] B. Kaliski. PKCS #1: RSA encryption version 1.5. RFC 2313 (Informational), March 1998. Obsoleted by RFC 2437.

[Kat08] Jonathan Katz. Bridging game theory and cryptography: recent results and future directions. In *Proceedings of the 5th conference on Theory of cryptography*, TCC'08, pages 251–272, Berlin, Heidelberg, 2008. Springer-Verlag.

[KM00] Steve Kremer and Olivier Markowitch. Optimistic non-repudiable information exchange. In *21th Symp. on Information Theory in the Benelux*, pages 139–146, Wassenaar, The Netherlands, May 2000.

[KMMS10] Barbara Kordy, Sjouke Mauw, Matthijs Melissen, and Patrick Schweitzer. Attack-defense trees and two-player binary zero-sum extensive form games are equivalent. In Tansu Alpcan, Levente Buttyán, and John S. Baras, editors, *GameSec*, volume 6442 of *Lecture Notes in Computer Science*, pages 245–256. Springer, 2010.

[KMRS10] Barbara Kordy, Sjouke Mauw, Sasa Radomirovic, and Patrick Schweitzer. Foundations of attack-defense trees. In Pierpaolo Degano, Sandro Etalle, and Joshua D. Guttman, editors, *Formal Aspects in Security and Trust*, volume 6561 of *Lecture Notes in Computer Science*, pages 80–95. Springer, 2010.

[KR02] Steve Kremer and Jean-François Raskin. Game analysis of abuse-free contract signing. In *CSFW*, pages 206–220. IEEE Computer Society, 2002.

[KR03] Steve Kremer and Jean-François Raskin. A game-based verification of non-repudiation and fair exchange protocols. *Journal of Computer Security*, 11(3):399–430, 2003.

[KR10] Simon Kramer and Andrey Rybalchenko. A multi-modal framework for achieving accountability in multi-agent systems. In *Proceedings of the ESSLLI-affiliated Workshop on Logics in Security*, 2010.

[KST10] Ralf Küsters, Henning Schnoor, and Tomasz Truderung. A formal definition of online abuse-freeness. In Sushil Jajodia and Jianying Zhou, editors, *SecureComm*, volume 50 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 484–497. Springer, 2010.

[KSW97] John Kelsey, Bruce Schneier, and David Wagner. Protocol interactions and the chosen protocol attack. In Bruce Christianson, Bruno Crispo, T. Mark A. Lomas, and Michael Roe, editors, *Security Protocols Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 91–104. Springer, 1997.

[KTV10] Ralf Küsters, Tomasz Truderung, and Andreas Vogt. Accountability: definition and relationship to verifiability. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM Conference on Computer and Communications Security*, pages 526–535. ACM, 2010.

[KV08] Francis Klay and Laurent Vigneron. Automatic methods for analyzing non-repudiation protocols with an active intruder. In Pierpaolo Degano, Joshua D. Guttman, and Fabio Martinelli, editors, *Formal Aspects in Security and Trust*, volume 5491 of *Lecture Notes in Computer Science*, pages 192–209. Springer, 2008.

[LMO06] François Laroussinie, Nicolas Markey, and Ghassan Oreiby. Model-checking timed ATL for durational concurrent game structures. In Asarin and Bouyer [AB06], pages 245–259.

[LNJ01] Peng Liu, Peng Ning, and Sushil Jajodia. Avoiding loss of fairness owing to failures in fair data exchange systems. *Decision Support Systems*, 31(3):337–350, 2001.

[Low96] Gavin Lowe. Breaking and fixing the needham-schroeder public-key protocol using fdr. *Software - Concepts and Tools*, 17(3):93–102, 1996.

[Low97] Gavin Lowe. A hierarchy of authentication specifications. In *Proceedings of the 10th IEEE workshop on Computer Security Foundations*, CSFW '97, pages 31–, Washington, DC, USA, 1997. IEEE Computer Society.

[LPZ11] Zhiyuan Liu, Jun Pang, and Chenyi Zhang. Verification of a key chain based ttp transparent cem protocol. *Electr. Notes Theor. Comput. Sci.*, 274:51–65, 2011.

[LQR09] Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. MCMAS: A model checker for the verification of multi-agent systems. In Ahmed Bouajjani and Oded Maler, editors, *CAV*, volume 5643 of *Lecture Notes in Computer Science*, pages 682–688. Springer, 2009.

[Mau04] Ueli M. Maurer. New approaches to digital evidence. *Proceedings of the IEEE*, 92(6):933–947, 2004.

[Mic10] Daniele Micciancio, editor. *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, volume 5978 of *Lecture Notes in Computer Science*. Springer, 2010.

[MK00] Olivier Markowitch and Steve Kremer. A multi-party optimistic non-repudiation protocol. In Dongho Won, editor, *ICISC*, volume 2015 of *Lecture Notes in Computer Science*, pages 109–122. Springer, 2000.

[MMPV11] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. Reasoning about strategies: On the model-checking problem. *CoRR*, abs/1112.6275, 2011.

[MO05] Sjouke Mauw and Martijn Oostdijk. Foundations of attack trees. In Dongho Won and Seungjoo Kim, editors, *ICISC*, volume 3935 of *Lecture Notes in Computer Science*, pages 186–198. Springer, 2005.

[MR08] Aybek Mukhamedov and Mark Dermot Ryan. Fair multi-party contract signing using private contract signatures. *Inf. Comput.*, 206(2-4):272–290, 2008.

[MRD09] Sjouke Mauw, Sasa Radomirovic, and Mohammad Torabi Dashti. Minimal message complexity of asynchronous multi-party contract signing. In *CSF*, pages 13–25. IEEE Computer Society, 2009.

[MS06] George J. Mailath and Larry Samuelson. Repeated games and reputations: Long-run relationships. Oxford University Press, 2006.

[Nak07] N. Nakanishi. Purely noncooperative farsighted stable set in an n-player Prisoners Dilemma. Technical Report 707, Kobe University, 2007.

[NM44] John Von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[ODSS04] Ole Kasper Olsen, Ole Martin Dahl, Torkjel Søndrol, and Fredrik Skarderud. Contract signing using pgp. Technical report, Gjøvik University College, NISlab, 2004.

[OR94] Martin J. Osborne and Ariel Rubinstein. *A course in game theory*. The MIT Press, Cambridge, USA, 1994. electronic edition.

[PE02] PKCS-Editor. PKCS#1 v2.1: RSA cryptography standard. Technical report, RSA Laboratories, April 2002.

[PG99] Henning Pagnia and Felix C. Gartner. On the impossibility of fair exchange without a trusted third party. Technical report, Darmstadt University of Technology, 1999.

[Pra] Best Practical. Request tracker. `http://www.bestpractical.com/rt/`. Accessed: 2013-09-11.

[Ram99] Blake Ramsdell. S/MIME version 3 message specification. Internet RFC 2633, June 1999.

[Ros96] A. William Roscoe. Intensional specifications of security protocols. In *CSFW*, pages 28–38. IEEE Computer Society, 1996.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[RSF+09] Martin Rehák, Eugen Staab, Volker Fusenig, Michal Pechoucek, Martin Grill, Jan Stiborek, Karel Bartos, and Thomas Engel. Runtime monitoring and dynamic reconfiguration for intrusion detection systems. In Engin Kirda, Somesh Jha, and Davide Balzarotti, editors, *RAID*, volume 5758 of *Lecture Notes in Computer Science*, pages 61–80. Springer, 2009.

[SaT] SaToSS. Vulnerability testing. `http://satoss.uni.lu/vulnerability-testing/`. Accessed: 2013-09-01.

[Sch98] Steve Schneider. Formal analysis of a non-repudiation protocol. In *CSFW*, pages 54–65, 1998.

[Sch99] Bruce Schneier. Attack trees - modeling security threats. *Dr. Dobb's Journal*, December 1999.

[Sch04a] Bruce Schneier. *Secrets and lies - digital security in a networked world: with new information about post-9/11 security*. Wiley, 2004.

[Sch04b] Pierre-Yves Schobbens. Alternating-time logic with imperfect recall. *Electr. Notes Theor. Comput. Sci.*, 85(2):82–93, 2004.

[Sch12] Henning Schnoor. Deciding epistemic and strategic properties of cryptographic protocols. In Sara Foresti, Moti Yung, and Fabio Martinelli, editors, *ESORICS*, volume 7459 of *Lecture Notes in Computer Science*, pages 91–108. Springer, 2012.

[SLB09] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[SM05] Akihiro Suzuki and Shigeo Muto. Farsighted stability in an n-person prisoner's dilemma. *Int. J. Game Theory*, 33(3):431–445, 2005.

[Syv98] Paul F. Syverson. Weakly secret bit commitment: Applications to lotteries and fair exchange. In *CSFW*, pages 2–13, 1998.

[THG99] F. Javier Thayer, Jonathan C. Herzog, and Joshua D. Guttman. Strand spaces: Proving security protocols correct. *Journal of Computer Security*, 7(1):191–230, 1999.

[Tim13] Washington Times. In classified cyberwar against iran, trail of stuxnet leak leads to white house. `http://www.washingtontimes.com/news/2013/aug/18/trail-of-stuxnet-cyberwar-leak-to-author-leads-to-`, 2013. Accessed: 2013-09-01.

[VD11] Ton Frederik Petrus Van Deursen. *Security of RFID protocols*. PhD thesis, Université du Luxembourg, Luxembourg, September 2011.

[vdHKL+10] Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, editors. *9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), Toronto, Canada, May 10-14, 2010, Volume 1-3*. IFAAMAS, 2010.

[vDMRV09] Ton van Deursen, Sjouke Mauw, Sasa Radomirovic, and Pim Vullers. Secure ownership and ownership transfer in RFID systems. In Backes and Ning [BN09], pages 637–654.

[WD09] Yanjing Wang and Francien Dechesne. On expressive power and class invariance. *CoRR*, abs/0905.4332, 2009.

[Wor13] Luxemburger Wort. 48.670 dossiers médicaux échappent au contrôle de l'Etat. `http://www.wort.lu/fr/view/4f60e9bae4b02f5ce8fa946d`, 2013. Accessed: 2013-09-01.

[ZG96] Jianying Zhou and Dieter Gollmann. Observations on non-repudiation. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT*, volume 1163 of *Lecture Notes in Computer Science*, pages 133–144. Springer, 1996.

[ZG97a] Jianying Zhou and Dieter Gollmann. An efficient non-repudiation protocol. In *CSFW* [DBL97], pages 126–132.

[ZG97b] Jianying Zhou and Dieter Gollmann. Evidence and non-repudiation. *Journal of Network and Computer Applications*, 20:267–281, 1997.

[ZG98] Jianying Zhou and Dieter Gollmann. Towards verification of non-repudiation protocols. In *1998 International Refinement Workshop and Formal Methods Pacific*, pages 370–380. Springer-Verlag, 1998.

[Zho10] Jianying Zhou. *Handbook of Financial Cryptography and Security (G. Rosenberg, ed.)*, chapter Non-repudiation, pages 82–108. Chapman and Hall/CRC, 2010.

[ZZPM09] Ying Zhang, Chenyi Zhang, Jun Pang, and Sjouke Mauw. Game-based verification of multi-party contract signing protocols. In Pierpaolo Degano and Joshua D. Guttman, editors, *Formal Aspects in Security and Trust*, volume 5983 of *Lecture Notes in Computer Science*, pages 186–200. Springer, 2009.

Publications

[EPIA] Wojciech Jamroga and Matthijs Melissen. Doubtful deviations and farsighted play. In Luis Antunes and Helena Sofia Pinto, editors, *Progress in Artificial Intelligence, 15th Portuguese Conference on Artificial Intelligence, EPIA 2011, Lisbon, Portugal, October 10-13, 2011. Proceedings*, volume 7026 of *Lecture Notes in Computer Science*, pages 506–520. Springer, 2011.

[STM] Wojciech Jamroga, Sjouke Mauw, and Matthijs Melissen. Fairness in Non-Repudiation Protocols. In Catherine Meadows and M. Carmen Fernández Gago, editors, *Proceedings of the 7th International Workshop on Security and Trust Management*, volume 7170 of *Lecture Notes in Computer Science*, pages 122–139. Springer, June 2012.

[PRIMA] Wojciech Jamroga, Matthijs Melissen, and Henning Schnoor. Incentives and rationality in security of interaction protocols. In *PRIMA 2013: Principles and Practice of Multi-Agent Systems. Proceedings*, 2013. In print.

[GameSec] Barbara Kordy, Sjouke Mauw, Matthijs Melissen, and Patrick Schweitzer. Attack-defense trees and two-player binary zero-sum extensive form games are equivalent. In Tansu Alpcan, Levente Buttyán, and John S. Baras, editors, *GameSec*, volume 6442 of *Lecture Notes in Computer Science*, pages 245–256. Springer, 2010.

[NSVKI] Matthijs Melissen. A solution to the emptiness problem for Lambek calculus and some of its extensions. In J H Janssen, A Van Wissen, and T Goosen, editors, *Proceedings of the 2nd NSVKI Student Conference*, Utrecht, June 2008.

[BNAIC] Matthijs Melissen. Lambek-Grishin Calculus Extended to Connectives of Arbitrary Arity. In Anton Nijholt, Maja Pantic, Mannes Poel, and Hendri Hondorp, editors, *Proceedings of the 20th Belgian-Netherlands Conference on Artificial Intelligence*, Enschede, October 2008.

[MSc] Matthijs Melissen. Lambek-Grishin Calculus Extended to Connectives of Arbitrary Arity. Master's thesis, Cognitive Artificial Intelligence, Universiteit Utrecht, May 2009.

[FG] Matthijs Melissen. The Generative Capacity of the Lambek-Grishin Calculus: A New Lower Bound. In Philippe de Groote, Markus Egg, and Laura Kallmeyer, editors, *Proceedings of Formal Grammar 2009*, volume 5591 of *LNCS*, pages 118–132. Springer, 2011.

Index of subjects

Curriculum Vitae

2013 – ...	Research fellow, University of Birmingham, United Kingdom.
2009 – 2013	Ph.D. student in Computer Science, University of Luxembourg, Luxembourg.
2007 – 2009	Master of Science in Cognitive Artificial Intelligence, Utrecht University, The Netherlands.
2004 – 2007	Bachelor of Science in Cognitive Artificial Intelligence, Utrecht University, The Netherlands.
1997 – 2004	Secondary education, Stedelijk Gymnasium Breda, The Netherlands.

Born on June 20, 1985, Breda, The Netherlands.