From the Institute of Information Systems
of the University of Lübeck
Director: Prof. Dr. Ralf Möller

PDT Logic:
A Probabilistic Doxastic Temporal Logic for Reasoning about Beliefs in Multi-agent Systems

Dissertation
for Fulfillment of
Requirements
for the Doctoral Degree
of the University of Lübeck

from the Department of Computer Sciences

Submitted by

Karsten Martiny
from Hamburg

Lübeck 2016

First referee: Prof. Dr. Ralf Möller

Second referee: Prof. Dr. Rüdiger Reischuk

Date of oral examination: October 4, 2016

Approved for printing. Lübeck, October 26, 2017

Abstract

This thesis presents *Probabilistic Doxastic Temporal (PDT) Logic*, a formalism to represent and reason about probabilistic beliefs and their temporal evolution in multi-agent systems. This formalism enables the quantification of agents' beliefs through probability intervals and incorporates an explicit notion of time.

Quantifying probabilistic knowledge through probability intervals instead of single probability values significantly eases the task of formally representing existing knowledge of a human domain expert. In most cases, a domain expert can give reasonable probability estimates of her knowledge, but will inevitably fail at giving correct precise numerical values on these probabilities. Thus, the use of probability intervals provides means to express probabilistic knowledge as precise as possible without enforcing unrealistic precision. The width of a probability interval can then give additional information about the certainty of a probability quantification. Naturally, a narrow interval is associated with a high certainty of the respective probability and vice versa, a wide interval is associated with low certainty.

In contrast to related work, PDT Logic employs an explicit notion of time and thereby facilitates the expression of richer temporal relations. Existing approaches of dynamic epistemic logics usually employ an implicit notion of time. This makes it impossible to reason about temporal relations that span over multiple time points.

Through the introduction of an appropriate syntax and semantics we discuss how beliefs in multi-agent systems can be formally represented and how their temporal evolutions can be analyzed. We analyze the complexity of decision problems in PDT Logic and develop sound and complete satisfiability checking algorithms. Possible applications of PDT Logic are indicated through the discussion of suitable examples.

Zusammenfassung

In der vorliegenden Arbeit wird *Probabilistic Doxastic Temporal (PDT) Logic* vorgestellt. PDT Logic ist ein Formalismus, um probabilistischen Glauben und dessen temporale Entwicklung in Multi-Agenten-Systemen darzustellen und daraus Schlüsse zu ziehen. Dieser Formalismus ermöglicht die Quantifizierung von Glaubenszuständen von Agenten durch Wahrscheinlichkeitsintervalle und verwendet eine explizite Darstellung der Zeit.

Die Quantifizierung von probabilistischem Wissen durch Wahrscheinlichkeitsintervalle statt einzelner Wahrscheinlichkeitswerte erleichtert menschlichen Experten die formale Darstellung existierenden Wissens signifikant. In den meisten Fällen kann ein Experte sinnvolle Wahrscheinlichkeitsabschätzungen seines Wissens liefern, während die Angabe von präzisen Werten zu unvermeidbaren Fehlern führt. Folglich ermöglicht die Verwendung von Wahrscheinlichkeitsintervallen die Darstellung von probabilistischem Wissen so genau wie möglich, ohne eine unrealistische Präzision zu erzwingen. Kleine Wahrscheinlichkeitsintervalle stellen naturgemäß einen hohen Grad an Sicherheit bezüglich des Wahrscheinlichkeitswertes dar und umgekehrt stellen große Wahrscheinlichkeitsintervalle eine geringe Sicherheit dar.

Im Gegensatz zu verwandten Arbeiten verwendet PDT Logic eine explizite Darstellung der Zeit und ermöglicht dadurch die Darstellung von komplexeren zeitlichen Zusammenhängen, während existierende Ansätze für dynamische epistemische Logiken in der Regel eine implizite Darstellung der Zeit verwenden. Hierdurch wird es unmöglich, Schlussfolgerungen aus temporalen Zusammenhängen zu ziehen, die sich über mehrere Zeitpunkte erstrecken.

Durch die Einführung geeigneter Syntax und Semantik wird diskutiert, wie Glauben in Multi-Agenten-Systemen formal dargestellt und dessen zeitliche Entwicklung analysiert werden kann. Die Komplexität von Entscheidungsproblemen für PDT Logic wird analysiert und vollständige und korrekte Entscheidungsalgorithmen werden entwickelt. Mögliche Anwendungen von PDT Logic werden durch die Diskussion geeigneter Beispiele aufgezeigt.

Contents

1. Introduction **1**
1.1. Research Objectives and Scientific Contributions 3
1.2. Dissemination Activities 4
1.3. Outline of the Dissertation 5

2. Related Work **7**

3. PDT Logic: Syntax and Semantics **11**
3.1. Syntax 11
3.2. Semantics 14
3.2.1. Possible Worlds 15
3.2.2. Threads 17
3.2.3. Kripke Structures 17
3.2.4. Subjective Posterior Temporal Probabilistic Interpretations . . . 20
3.2.5. Frequency Functions 23
3.2.6. Semantics of the Belief Operator 25
3.3. Evolution over Time 28
3.3.1. Evolution of Probabilistic Interpretations 30
3.3.2. Evolution of Beliefs 32
3.4. Concluding Remarks 37

4. Satisfiability Checking for PDT Logic **39**
4.1. A Model Checking Algorithm 40
4.2. A Compact Problem Specification 42
4.2.1. Identification of Key Parameters from a Set of Belief Formulae . 43
4.3. Representing the Satisfiability Problem as a Linear Program 47
4.3.1. Representation of Subjective Posterior Probabilities 48
4.3.2. Extracting Linear Constraints from Belief Formulae 52
4.3.3. Transformation into a Disjunctive Program 57
4.3.4. Transformation into a 0-1 Mixed Integer Linear Program 59

4.4. Prior Constraints on Possible Threads 61
4.4.1. A Taxonomy of Belief Formulae 61
4.4.2. Constraining Possible Worlds at Individual Time Points 69
4.5. Representative Threads 73
4.5.1. Generating Representative Threads 74
4.5.2. A Thread Generation Example 81
4.5.3. Properties of the Representative Thread Generation 85
4.6. Concluding Remarks 88

5. Analyzing Attackers' Beliefs: An Application Example in Cyber Security Domains 89
5.1. Threat Analysis with Attack Graphs 89
5.2. Formalizing Agents' Beliefs 90
5.2.1. Considerations on the Target Domain 90
5.2.2. An Exemplary Domain Model 92
5.2.3. An Analysis of Belief Evolutions 95
5.3. Concluding Remarks 96

6. Abduction in PDT Logic 97
6.1. Formalizing the Abduction Problem in PDT Logic 99
6.2. The Hypothesis Space 102
6.3. The Abduction Process 107
6.4. Concluding Remarks 111

7. Extending Time Frames to Markovian Streams 113
7.1. Substituting Threads with Infinite Streams of Possible Worlds 114
7.2. Time Windows 116
7.3. Prior Probabilities for Threads within Time Windows 117
7.4. A Stream Example 119
7.5. Concluding Remarks 121

8. Conclusion 123

A. Complexity Proof for Abduction in APT Logic 127

List of Figures 131

Bibliography 133

Chapter 1

Introduction

Logical analysis of knowledge and belief has been an active topic of research in diverse fields such as philosophy [Hin62], economics [Aum76], game theory [Har67, Har68a, Har68b], and computer science [FHMV95]. Numerous extensions to modal epistemic logic have been made to reason about knowledge in multi-agent settings [FHMV95, BM04], to add probabilistic knowledge [FH94, CEMS08], and to analyze the dynamic evolution of knowledge [vDvdHK07].

In most realistic scenarios, an agent has only incomplete and inaccurate information about the actual state of the world, and thus considers several different worlds as actually being possible. As it receives new information (e.g., it observes some facts that currently hold), it has to update its beliefs about possible worlds such that they are consistent with this new information. These updates can for example result in regarding some (previously considered possible) worlds as impossible or judging some worlds to be more likely than before. Thus, in addition to analyzing the set of worlds an agent believes to be possible, it is also useful to quantify these beliefs in terms of probabilities. This provides means to specify fine-grained distinctions between the range of worlds that an agent considers possible but highly unlikely, and worlds that seem to be almost certainly the actual world.

When multiple agents are involved in such a setting, an agent may not only have varying beliefs regarding the facts of the actual world, but also regarding the beliefs of other agents. In many scenarios, the actions of one agent will not only depend on its belief in ontic facts (i.e., facts of the actual world), but also on its beliefs in some other agent's beliefs.

To illustrate how reasoning about other agents' beliefs can yield significant advantages in practical scenarios, we start with the following informal description of an application from the cyber security domain: Suppose that an adversary is trying to break into a computer system. This is usually done with an attack graph to detect and exploit potential vulnerabilities of the system. An attack graph specifies a set of paths (i.e., sequences of actions) to carry out an attack. Several paths of the attack graph might be used in parallel, potentially by different agents (for instance, a number of infected computers controlled by a botnet). Usually, attack patterns specified by one attack graph are used multiple times, which has two important ramifications: the adversary will learn from experience which of the paths yield a high probability of successfully breaking into a system. Defenders in turn will be able to gain knowledge of the attack graph through the repeated observation of certain patterns. Thus, when a system is under attack, the defender will have beliefs about both the chosen attack paths and the adversary's belief regarding the success of the respective path. Thus, the defender can choose countermeasures effectively by reacting only on paths where these nested beliefs are high and which indeed pose a threat according the system's mission impact model.

This thesis introduces *Probabilistic Doxastic Temporal (PDT) Logic* to formalize reasoning about such beliefs in multi-agent settings. PDT Logic builds upon recent work on Annotated Probabilistic Temporal (APT) Logic [SPSS11, SSS12] and provides a formalism which enables representing and reasoning about dynamically changing quantified temporal multi-agent beliefs through probability intervals and incorporates a subset of epistemic actions [BM04]. Using concepts from APT Logic as a semantic foundation, PDT Logic merges work on epistemic logic with recent work on temporal logic [SPSS11, SSS12]. Apart from reasoning about imprecise probabilities, this introduces the temporal concept of frequency functions into epistemic temporal logic.

Quantifying probabilistic knowledge through probability intervals instead of single probability values yields two main advantages. On the one hand, using probability intervals significantly eases the task of formally representing existing knowledge of a human domain expert. In most cases, a domain expert can give reasonable probability estimates of her knowledge, but will inevitably fail at giving correct precise numerical values on these probabilities. Consider for instance a weather forecast: most people find it easy to give coarse probabilistic quantifications such as "the chance of rain is high", while virtually nobody could quantify this through an exact numerical value. Employing exact numerical values in a formal representation would then inevitably introduce errors in the probability model. Thus, the use of probability intervals provides means to express probabilistic knowledge as precisely as possible without enforcing unrealistic precision. On the other hand, there are many scenarios where probabilities (and even rough estimates of them) are simply unavailable, while bounds on these values may be known. To illustrate this, consider the scenario described in [Ell61]:

Example 1.1 (The Ellsberg paradox [Ell61])**.** Imagine an urn known to contain 30 red balls and 60 black and yellow balls, the latter in unknown proportion. One ball is to be drawn at random from the urn; the following actions are considered: Action I is "a bet on red", II is "a bet on black".

Now, it is easy to see that any rational agent would believe that action I will be successful with a probability of $1/3$. For action II, no such quantification is possible because the respective probability is unknown. Yet omitting any probabilistic information about action II altogether would ignore some available information about the unknown probability value, namely that it is somewhere between 0 and $2/3$. This example exhibits two different types of uncertainty: the former action is subject to *risk*, i.e., the outcome is unknown, but occurs with known probability, while the later action is subject to *ambiguity* (also known as *Knightian uncertainty*), where the probability is unknown [Bra15]. Through probability intervals, PDT Logic is able to work with such imprecise probabilities. The width of a probability interval can then give additional information about the certainty of a probability quantification. Naturally, a narrow interval is associated with a high certainty of the respective probability and vice versa, a wide interval is associated with low certainty.

PDT Logic employs an explicit notion of time and thereby facilitates the expression of richer temporal relations. This allows for the analysis of temporal doxastic problems beyond the scope of previous work. The resulting framework provides means to reason about the temporal evolution of beliefs in multi-agent systems. Two different applications of this framework are possible: First, any agent of the respective multi-agent system can employ this framework online during a run of the system to reason about its own beliefs. By analyzing nested beliefs as introduced above, this gives an agent also means to reason about probable evolutions of other agents' belief states. Second, this framework can be used offline by an external observer to analyze whether desired evolutions of a given system are possible.

1.1. Research Objectives and Scientific Contributions

The main goal of the research presented in this thesis is the development of a formalism to represent beliefs and their temporal evolutions in multi-agent systems. In detail, the objective of this work is to provide a formalism with the following features:

- Agents' beliefs can be be quantified with probability intervals. The use of probabilities enables a formal representation of varying strengths of beliefs, while the use of probability intervals facilitates imprecise probabilities and therefore gives a more natural way of specifying existing knowledge.
- Beliefs can be expressed about specific facts of the world, about temporal evolutions of the world, and about beliefs of other agents.
- Various temporal concepts can be expressed using an explicit notion of time.
- Upon receiving new information, agents should update their beliefs such that this information is correctly incorporated.

Using APT Logic as a semantic foundation, the development of *Probabilistic Doxastic Temporal (PDT) Logic* in this work provides a bridge between recent work on temporal logic and established areas of epistemic logic. As a result, we obtain a novel formalism to represent beliefs in multi-agent systems that meets the above criteria.

The major contributions of this thesis are as follows:

- Inspired by recent contributions from temporal logics—namely APT Logic—we introduce the novel formalism *PDT Logic* to represent and reason about evolving beliefs with imprecise probabilities in multi-agent systems. For this formalism we define a comprehensive syntax and give a suitable well-defined formal semantics. While we define our formalism with respect to finite time frames, we also show how this can be applied to infinite time domains under certain conditions.
- We provide an observation model to represent agents' perception of new information over time. Based on these observations we derive an update rule such that agents' beliefs evolve in accordance with any new information that they may receive over time.
- We discuss alternative approaches of specifying problems in PDT Logic, each with specific merits and drawbacks. For these approaches, we show that the respective satisfiability problem is decidable and provide suitable decision procedures.
- We show how abductive reasoning can be be formalized in PDT Logic to formally reason about possible actions that can induce a desired belief state for some agents of our multi-agent system.

1.2. Dissemination Activities

Various parts of this thesis have been published previously to disseminate research results at different stages. The following list provides an overview of dissemination activities in chronological order.

- Karsten Martiny and Ralf Möller:
  PDT Logic for Stream Reasoning in Multi-agent Systems
  in
  6th International Symposium on Symbolic Computation in Software Science (SCSS), Tunis, Tunisia, 2014

- Karsten Martiny and Ralf Möller:
  A Probabilistic Doxastic Temporal Logic for Reasoning about Beliefs in Multi-agent Systems
  in
  7th International Conference on Agents and Artificial Intelligence (ICAART), Lisbon, Portugal, 2015
- Karsten Martiny, Alexander Motzek, and Ralf Möller:
  Formalizing Agents' Beliefs for Cyber-Security Defense Strategy Planning
  in
  8th International Conference on Computational Intelligence in Security for Information Systems, Burgos, Spain, 2015
- Karsten Martiny and Ralf Möller:
  Abduction in PDT Logic
  in
  28th Australasian Conference on Artificial Intelligence (AI), Canberra, Australia, 2015
- Karsten Martiny and Ralf Möller:
  PDT Logic: A Probabilistic Doxastic Temporal Logic for Reasoning about Beliefs in Multi-agent Systems,
  in
  Journal of Artificial Intelligence Research (JAIR), Volume 57, pages 39-112, September 2016
- Karsten Martiny and Ralf Möller:
  Reasoning about Imprecise Beliefs in Multi-Agent Systems,
  accepted for publication in
  KI Zeitschrift - Special Issue on Challenges for Reasoning under Uncertainty, Inconsistency, Vagueness, and Preferences

1.3. Outline of the Dissertation

Research on formal representations of knowledge and belief has led to an ample variety of scientific contributions over the last decades. These contributions can be broadly classified into four different—but overlapping—subfields of epistemic logic: multi-agent epistemic logic, probabilistic epistemic logic, epistemic temporal logic, and dynamic epistemic logic. To put this thesis in its proper perspective, Chapter 2 gives a review of major contributions in each of these four areas, as well as a summary of APT Logic.

Chapter 3 introduces the formal foundations of PDT Logic. We introduce the syntax of our formalism, discuss our finite model of time, and provide the formal semantics for our approach. Moreover, we define an update rule that enables agents to update their beliefs upon receiving new information, such that this new information is correctly integrated into their belief state.

In Chapter 4 we discuss alternative approaches of modeling problems in PDT Logic, show for these approaches that the corresponding satisfiability problems are decidable, and provide suitable decision procedures. In selected problem domains, it is possible to specify problems through an explicit enumeration of all possible evolutions over time and give according prior probabilities for each possible evolution. We show that if such a specification is given, the time complexity of the satisfiability problem is polynomial in the number of specified belief operators. By transforming the PDT Logic satisfiability problem to a linear program, we show that satisfiability can be decided even if no precise prior probabilities are given. However, this approach still relies on an explicit specification of all possible evolutions in a problem domain. Since creating such a specification is a very tedious task and unreasonable in most cases, we discuss an alternative problem specification, where it suffices for a domain expert to specify her knowledge through sets of facts and rules. While this significantly simplifies the modeling task, we show that this comes at the cost of a significantly higher computational complexity. Although the formal semantics defined in Chapter 3 is based on precise probability values, treatment of the latter problem specification shows that satisfiability in PDT Logic can be decided even if only imprecise probabilities are given.

A possible application of PDT Logic to cyber security scenarios is discussed in Chapter 5. We show how the established concept of using attack graphs to analyze security threats can be adapted to PDT Logic. This extends current analysis capabilities to consider attackers' belief states when planning a suitable defense strategy.

Chapter 5 already shows informally that in many scenarios it is useful to reason about possible actions to induce a desired belief state for some agent. Chapter 6 addresses this problem formally by introducing abduction to PDT Logic. We define a formal account of the abduction problem, show how potential actions to achieve the goal can be obtained automatically, and derive an abduction procedure.

In its basic form, PDT Logic is only suitable for problems that can be represented through finite time frames. In Chapter 7 we show how this restriction can be removed by replacing finite time frames with infinite Markovian streams. Within these streams, finite-length time windows can be placed arbitrarily to reason about the current state and near future of the represented problem. We show how these time windows together with an adapted notion of prior probabilities provide a bridge from our previously introduced concepts to domains with infinite streams.

Chapter 8 concludes this work with a summary of the main achievements. Moreover, we present promising directions for future work.

Chapter 2

Related Work

Approaches to formalize reasoning about knowledge and belief date back to Hintikka's work on epistemic logic [Hin62]. Hintikka proposed to represent knowledge through sets of states or worlds, together with a binary relation for every agent, to determine which worlds are indistinguishable for an agent. This approach has sparked multiple branches of research on epistemic logic, which are still active topics of research today. These branches of research can be broadly classified into four (not mutually exclusive) areas that are relevant for our work: multi-agent epistemic logic, probabilistic epistemic logic, epistemic temporal logic, and dynamic epistemic logic.[1] In the following, we give an overview of the key contributions in each area and discuss existing approaches that merge these fields of research.

Early research on epistemic logic culminated in the influential work *Reasoning about Knowledge* [FHMV95], which provides a unified presentation of various preceding contributions on epistemic logic. This work uses a so-called interpreted systems approach to represent knowledge in multi-agent systems, where time is represented through *runs*. A run is a sequence of a system's global states and it thus identifies the state of a system for every time point. Among other contributions, this work provides notions for multi-agent epistemic modalities such as nested knowledge, distributed knowledge, and common knowledge.

Several works have extended epistemic logic to represent dynamic evolutions of knowledge. This direction of research is known as *Dynamic Epistemic Logic (DEL)*. The first work to formally analyze the dynamics of knowledge is [Pla89] (reprinted in [Pla07]). In this contribution, Plaza introduces *public communication* events (now commonly known as *public announcements*) to analyze the dynamic evolution of knowledge

[1]To simplify the following discussion, we do not explicitly distinguish between epistemic and doxastic logics in this section, but use "epistemic" as a general term. Strictly speaking, epistemic formalisms deal with knowledge, while doxastic formalisms deal with beliefs. The usual axiomatic definition of knowledge in the literature uses the *Truth Axiom*, which stipulates that an agent can only know true facts. Omitting this axiom then leads to the notion of belief. Even though not unanimously accepted (cf. e.g., [HSS09]), this axiom is usually considered as the key distinction between knowledge and belief.

in groups upon truthful public announcements of facts to a group of agents. Independently from [Pla89] a related approach for a public announcement logic was proposed in [GG97]. In [BMS98] and [BM04] the dynamic approach to epistemic logic is generalized to incorporate a variety of complex epistemic actions. Here, epistemic updates themselves are represented through Kripke models. This extends dynamic epistemic logic to represent a variety of additional epistemic actions such as private group announcements (i.e., announcements where agents outside of the receiving group are unaware of this announcement), lies (i.e., untruthful announcements), and combinations thereof. In PDT Logic, we use public and private group announcements, but we assume that all announcements are truthful. A thorough treatment of dynamic epistemic logic can be found in [vDvdHK07]. A recent overview of this field can be found in [vE14].

An alternative approach of modeling the evolution of knowledge is to combine epistemic logic with some temporal system. One example for this are the aforementioned interpreted systems from [FHMV95]. Another approach of modeling temporal aspects in epistemic logic was proposed by [PR03]. This approach is known as *Epistemic Temporal Logic (ETL)*. Here, possible situations are represented through sets of histories, with local histories for every agent, which represent the respective agent's previous observations. Based on these histories, knowledge based semantics of messages are defined, and it is shown that messages can vary in meaning, depending on the respective context of the message's receiver. The temporal model we employ in PDT Logic is closely related to epistemic temporal logic. Instead of specifying local histories for every agent, we define the semantics of PDT Logic with respect to a global history. However, the local contexts in the sense of ETL can easily be extracted from the global history by filtering this history for the respective agents' observations.

The traditional work on epistemic logic discussed so far does not allow to quantify an agent's degree of belief in certain facts; it can only be specified whether an agent does or does not know (resp. believe) some fact. To remove this limitation, several approaches have been proposed to combine logics of knowledge and belief with probabilistic quantifications. Fagin and Halpern laid the foundation for this combination in their seminal paper [FH94]. They define a belief operator to quantify lower bounds on the probabilities that an agent assigns to a formula. This is modeled by associating a probability space with each state and each agent. In their framework, it is generally not guaranteed that formulae define measurable sets, but they present some properties that can guarantee the measurability of such sets. In contrast, the semantics defined for PDT Logic always produces events with measurable probabilities. A special case of the framework introduced in [FH94] is presented in [MK00]. Just as in PDT Logic, in this formalism it is assumed (i) that there exists a common prior probability distribution over the set of worlds and (ii) that each agent's local probability distribution at some world is derived from the global distribution conditioned on the respective set of worlds the agent considers possible. The additional feature in [MK00] is that models are represented as Bayesian networks to find the probabilities of defined formulae. In [vdH97], the logic $P_F D$ is introduced, and later extended in [dCFFvdH08]. Like [FH94], this framework introduces an operator to quantify the lower bounds of probabilistic beliefs. Probabilistic values in this work are semantically restricted to a finite base set of probability values, yielding a logically compact framework that enables efficient implementations.

A variety of approaches have been proposed to extend probabilistic epistemic logics to dynamic frameworks: [Koo03] restricts the probabilistic epistemic logic from [FH94] to finite settings and combines it with the dynamic epistemic logic from [GG97] to create *Probabilistic Dynamic Epistemic Logic (PDEL)*. This work analyzes the effects on probabilistic beliefs upon public announcements. As this framework is based on dynamic epistemic logic, it does not have capabilities to represent temporal relationships; features regarding the past cannot be expressed at all, and features regarding the future can only be expressed to a limited extent as the result of certain actions. In [vB03] this framework is extended to analyze the results of various epistemic actions as described in [BMS98]. Another extension to this framework is proposed in [vBGK09], where different sources of probabilities are distinguished. A simplification of this approach is presented in [vES14]. This paper distinguishes itself from the above work on probabilistic epistemic logic in that certainty is equated with knowledge. Other works make an explicit distinction between belief with probability 1 and knowledge. The difference between these two concepts is often illustrated with repeatedly throwing a fair coin: the event that the coin shows head at least once is 1 for an infinite number of repetitions. Yet no agent can *know* in this example that the coin will eventually show head. As PDT Logic works only with countable models in finite time frames, we can adopt the view from [vES14] and consider certainty and knowledge as equivalent in our models. Deviating from these approaches to extend epistemic logic with probabilities, PDT Logic provides a belief operator with probability interval quantifications, so that both lower and upper bounds on the probability values can be specified explicitly. This provides a natural means to represent imprecise probabilities as discussed in the introduction.

In dynamic epistemic logic, it is only possible to reason about step-wise changes in the future. In order to reason about temporal relations, [Sac08] extends the update mechanism of dynamic epistemic logic with temporal operators, namely previous-time and next-time operators. In [Sac09], this approach is extended to probabilistic frameworks by augmenting the work on probabilistic dynamic epistemic logic [Koo03] with a previous-time operator and the ability to reason about continuous probabilities. These approaches enrich dynamic epistemic logic with the ability to reason about events in the past. In [vBGHP09], a systematic and precise comparison between ETL (called TEL in [vBGHP09]) and DEL is given and it is shown how these approaches can be merged into a single framework.

[SPSS11] and [SSS12] introduce APT Logic, a framework to represent probabilistic temporal evolutions of worlds in threads. APT Logic assigns prior probabilities to every thread and uses these probabilities to determine probabilities of events occurring in specific threads. To represent temporal relationships between events, APT Logic introduces the concept of frequency functions. We utilize the approach of APT Logic to create a doxastic multi-agent framework that supports explicit reasoning about temporal relationships through the adoption of frequency functions. While the explicit notion of time in our formalism increases the complexity of decision problems, it significantly enhances the expressibility of temporal relations. For instance, in contrast to all approaches with implicit representations of time, in PDT Logic we are able to specify that events occur within a certain time interval (cf. the introduction of frequency functions below).

PDT Logic: Syntax and Semantics

In this chapter, we discuss how beliefs in multi-agent systems can be formalized. We start with defining the syntax of PDT Logic, discuss the employed model of time, and provide a formal semantics. The proposed formalism enables the expression of different types of beliefs and can quantify these beliefs using imprecise probabilities. By introducing a suitable update rule we show how agents' beliefs evolve over time and how agents can update their beliefs such that new information is correctly integrated into their belief state.

3.1. Syntax

We assume the existence of a function-free and quantifier-free fragment of first order logic[1] language $\mathcal{L}$ with finite sets of constant symbols $\mathcal{L}_{cons}$ and predicate symbols $\mathcal{L}_{pred}$, and an infinite set of variable symbols $\mathcal{L}_{var}$. Every predicate symbol $p \in \mathcal{L}_{pred}$ has an *arity*. A *term* is any member of the set $\mathcal{L}_{cons} \cup \mathcal{L}_{var}$. A term is called a *ground term* if it is a member of $\mathcal{L}_{cons}$. If $t_1, .., t_n$ are (ground) terms, and p is a predicate symbol in $\mathcal{L}_{pred}$ with arity n, then $p(t_1, ..., t_n)$ is a (ground) atom. If a is a (ground) atom, then a and $\neg a$ are (ground) *literals*. The former is called a *positive literal*, the latter is called a *negative literal*. The set of all ground literals is denoted by $\mathcal{L}_{lit}$. As usual, $\mathcal{B}$ denotes the Herbrand Base of $\mathcal{L}$, i.e., the set of all ground atoms that can be formed through from $\mathcal{L}_{pred}$ and $\mathcal{L}_{cons}$.

Time is modeled in discrete steps and we assume that all agents reason about an arbitrarily large, but fixed-size window of time. The set of time points is given by $\tau = \{1, ..., t_{max}\}$. The set of agents is denoted by $\mathcal{A}$. Again, we assume that this set may be arbitrarily large, but of finite size. To describe what agents observe, we define observation atoms as follows.

[1]We use a first order structure for our language definition to have a syntactically convenient way of representing observations. Apart from this, propositional logic could be used as a base language.

Definition 3.1 (Observation atoms)**.** For any non-empty group of agents $\mathcal{G} \subseteq \mathcal{A}$ and ground literal $l \in \mathcal{L}_{lit}$, $Obs_{\mathcal{G}}(l)$ is an *observation atom.* The set of all observation atoms is denoted by $\mathcal{L}_{obs}$.

Intuitively, the meaning of a statement of the form $Obs_{\mathcal{G}}(l)$ is that all agents in the group $\mathcal{G}$ observe that the fact l holds. Note that l may be a negative literal and therefore we can explicitly specify observations of certain facts being false (such as "it is not raining"). We assume that the agents in $\mathcal{G}$ not only observe that l holds, but that each agent in $\mathcal{G}$ is also aware that all other agents in $\mathcal{G}$ make the same observation. In the line of [BM04], observations can be viewed as the effects of private group announcements of a fact l to a group $\mathcal{G}$ (i.e., l becomes common knowledge *within* $\mathcal{G}$, while all agents *outside* of $\mathcal{G}$ remain entirely oblivious of the observation): it represents an epistemic action, i.e., it alters the belief states of all agents in $\mathcal{G}$ (as formally defined below), but does not influence the ontic facts of the respective world.

Definition 3.2 (Formulae)**.** Both atoms and observation atoms are formulae. If F and G are formulae, then $F \wedge G$, $F \vee G$, and $\neg F$ are formulae. A formula is ground if all atoms of the formula are ground.

Example 3.1 (Coin toss)**.** Consider two agents $1, 2$ and a coin that is tossed. The event that the coin lands heads is denoted by the primitive proposition $Head$, and accordingly, the coin lands tails is denoted by $\neg Head$. Let us assume that the coin actually lands heads. Then, all sets of possible observations in this scenario are $\{Obs_{\{1\}}(Head)\}$, $\{Obs_{\{2\}}(Head)\}$, $\{Obs_{\{1\}}(Head), Obs_{\{2\}}(Head)\}$, $\{Obs_{\{1,2\}}(Head)\}$.

Note that there is a difference between the third and the fourth set: in the former scenario, both agents observe the outcome of the coin throw but both are unaware that the other agent actually made the same observation. In the latter scenario, both agents observe the outcome and are aware that the other agent observes the same. Since we do not allow for nesting of observations (i.e., expressions such as $Obs_{\mathcal{G}_1}(Obs_{\mathcal{G}_2}(l))$) in PDT Logic, only a subset of the epistemic actions discussed in [BM04] can be represented in our formalism. While this limits the expressivity of epistemic actions to some extent, we can ensure that the resulting set of possible observations $\mathcal{L}_{obs}$ is always finite and therefore we can show that PDT Logic is decidable (as shown in Chapter 4). Further, note that the formal concept of observations is not limited to express passive acts of observing facts, but can instead be used to model a wide range of actions: for instance, in the above example one could also use $Obs_{\{1,2\}}(Head)$ to model the act of one agent telling the other about the outcome of the coin throw—the ramifications of the communication act are exactly the same as they would be in a shared observation (assuming that agents do not lie).

To express temporal relationships, we define temporal rules following the approach of APT rules from [SPSS11]. The definition of temporal rules already relies on the concept of frequency functions, even though these are defined in the next section. We still introduce temporal rules now to enable a clearly separated presentation of syntax and semantics of PDT Logic.

Definition 3.3 (Temporal rules)**.** Let F, G be two ground formulae, Δt a time interval, and fr a name for a frequency function (as defined below in Section 3.2.5). Then $r^{\mathsf{fr}}_{\Delta t}(F, G)$ is called a temporal rule.

Frequency functions provide information about temporal connections between events. The meaning of an expression $r^{\mathsf{fr}}_{\Delta t}(F, G)$ is to be understood as "F is followed by G in Δt time units w.r.t. frequency function fr". Frequency functions enable the specification of various types of temporal relations. For example, they can be used to determine how often F is followed by G *within* Δt time units or how often F is followed by G *exactly after* Δt time units. The usage of fr in the syntax of temporal rules is used to specify a set of possible names for the employed types of frequency function.

Now, we can define the belief operator $B^{\ell,u}_{i,t'}$ to express agents' beliefs. Intuitively, $B^{\ell,u}_{i,t'}(\varphi)$ means that at time t', agent i believes that some fact φ is true with a probability $p \in [\ell, u]$. Particularly, the intuitive meaning of belief in a temporal rule is that agent i believes that G will hold according to $r^{\mathsf{fr}}_{\Delta t}(F, G)$, given that F holds at some time point. We call the probability interval $[\ell, u]$ the *quantification* of agent i's belief. We use F_t to denote that formula F holds at time t and, accordingly, $Obs_{\mathcal{G}}(l)_t$ to denote that an observation $Obs_{\mathcal{G}}(l)$ occurs at time t. We call these expressions time-stamped formulae and time-stamped observation atoms, respectively.

Definition 3.4 (Belief formulae)**.** Let i be an agent, t' a time point, and $[\ell, u] \subseteq [0, 1]$. Then, *belief formulae* are inductively defined as follows:

1. If F is a ground formula and t is a time point, then $B^{\ell,u}_{i,t'}(F_t)$ is a belief formula.
2. If $r^{\mathsf{fr}}_{\Delta t}(F, G)$ is a temporal rule, then $B^{\ell,u}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(F, G))$ is a belief formula.
3. If $\mathscr{F}$ and $\mathscr{G}$ are belief formulae, then so are $B^{\ell,u}_{i,t'}(\mathscr{F})$, $\mathscr{F} \wedge \mathscr{G}$, $\mathscr{F} \vee \mathscr{G}$, and $\neg\mathscr{F}$.

For a belief $B^{\ell,u}_{i,t'}(\varphi)$ about something, we call φ the belief object. Belief operators are the atomic elements in PDT Logic, i.e., any expression $B^{\ell,u}_{i,t'}(\varphi)$ (including possibly nested belief formulae) is called an atom. We use script fonts (e.g., $\mathscr{F}$) to distinguish belief formulae from standard formulae. Note that we can have both ontic facts and observation atoms as standard formulae (cf. Definition 3.2) and therefore agents can also have beliefs about possible observations.

The use of probability intervals $[\ell, u]$ provides an option to represent imprecise probabilities [Bra15]: When using imprecise probabilities, it is usually assumed that the degree of belief in some proposition is not represented using a single probability function $p(\cdot)$, but instead through a set P of such functions. Then, the belief state $P(\varphi)$ in a proposition φ is represented through the set

$$P(\varphi) = \{p(\varphi) : p \in P\}.$$

For this set of probabilities $P(\varphi)$, so-called *lower* and *upper envelopes* are defined as $\underline{P}(\varphi) = \inf P(\varphi)$ and $\overline{P}(\varphi) = \sup P(\varphi)$, respectively. The belief quantifications in our belief operator represent such imprecise probabilities and the ℓ and u values of the probabilistic belief can be considered as the lower and upper envelopes $\underline{P}$ and $\overline{P}$ of the respective imprecise probability.

Remark 3.1. We decided to index both the belief operators $B^{\ell,u}_{i,t'}(\varphi)$ and facts F_t appearing as belief objects φ with time stamps to allow for a concise representation of temporal relations. Alternatively, one could use the more traditional approach (cf. [Sac09] for example) and introduce previous-time and next-time operators into the language to express temporal relationships between t and t' in $B^{\ell,u}_{i,t'}(F_t)$. Then, we could also omit the temporal index t' of the belief operator and instead evaluate whether the belief holds at time t' of the model. However, these are merely syntactic considerations that do not impact the underlying formalism. Thus we decided to encode time explicitly into the belief operators to avoid the introduction of additional temporal operators. Moreover, belief operators can also be used to express general temporal relationships of the modeled domain. We will illustrate this point in detail in Chapter 4.

3.2. Semantics

In this section, we will provide a formal semantics for PDT Logic that captures the intuitions explained above. To ease understanding of the presentation, we start with the introduction of an example, to which we will return repeatedly when introducing the various concepts of the semantics. For an illustration of our formalism's features, we use a simplified exemplary domain. While the practical use of this example is somewhat limited, it serves to illustrate how PDT Logic can be applied, and especially how the analysis of multi-agent beliefs can yield valuable information when deciding on meaningful actions. The resulting insights can then be easily applied to more sophisticated domains, as shown in subsequent chapters.

Example 3.2 (Trains)**.** Let Alice and Bob be two agents living in two different cities C_A and C_B, respectively. Suppose that Alice wants to take a train to visit Bob. Unfortunately, there is no direct connection between cities C_A and C_B, so Alice has to change trains at a third city C_C. We assume that train T_1 connects

C_A and C_C, and train T_2 connects C_C and C_B. Both trains usually require 2 time units for their trip, but they might be running late and arrive one time unit later than scheduled. Alice requires one time unit to change trains at city C_C. If T_1 runs on time, she has a direct connection to T_2, otherwise she has to wait for two time units until the next train T_2 leaves at city C_C. If a train is running late, she can call Bob to let him know. These calls can be modeled as shared observations between Alice and Bob. For instance, if Alice wants to tell Bob that train T_1 is running late (i.e., T_1 does not arrive at C_C at the expected time), this can be modeled as $Obs_{\{AB\}}(\neg at(T_1, C_C))$ at the expected arrival time.

3.2.1. Possible Worlds

Ontic facts and corresponding observations (e.g., as described in the above example) form *worlds* (or *states* in the terminology of [FHMV95]). A world ω consists of a set of ground atoms and a set of observation atoms, i.e., $\omega \in 2^{\mathcal{B} \cup \mathcal{L}_{obs}}$.[2] We use $a \in \omega$ and $Obs_{\mathcal{G}}(l) \in \omega$ to denote that an atom a, resp. observation atom $Obs_{\mathcal{G}}(l)$, holds in world ω. Since agents can only observe facts that actually hold in the respective world, we can define admissibility conditions of worlds w.r.t. the set of observations:

Definition 3.5 (Admissible worlds)**.** A world ω is admissible, iff for every observation atom $Obs_{\mathcal{G}}(l) \in \omega$

1. the observed fact holds, i.e., $x \in \omega$ if l is a positive literal x, and $x \notin \omega$ if l is a negative literal $\neg x$, and
2. for every subgroup $\mathcal{G}' \subset \mathcal{G}$, $Obs_{\mathcal{G}'}(l) \in \omega$.

We use $adm(\omega)$ to denote that a world ω is admissible.

The set of all possible worlds is denoted by Ω and the set of admissible worlds by $\hat{\Omega}$. For the following discussion in this chapter we assume that some specification of $\hat{\Omega}$ is given. While it is possible to employ the usual definition of Ω as the set of all combinations of ground atoms and observation atoms ($\Omega = 2^{\mathcal{B} \cup \mathcal{L}_{obs}}$), and $\hat{\Omega}$ as the maximum subset of Ω complying with Definition 3.5, this usually contains a vast number of worlds which are blatantly impossible according to the respective problem modeled.

[2] Most formalisms in epistemic logic do not encode facts directly into the worlds, but instead use a set of named states $s_1, s_2, \ldots$ and some valuation function $\pi(s_i)$ to determine which facts hold in world s_i (cf. [FHMV95]). This is mainly done to obtain the option of having multiple worlds s_i, s_j where the same facts hold (i.e., $\pi(s_i) = \pi(s_j)$), but the knowledge states of the agents differ. As described below, in PDT Logic worlds appear within threads, and thus it is possible that worlds with the same valuation appear at some time point in multiple threads. Thus, in our formalism we can encode facts directly into the possible worlds and save the valuation function without limiting the epistemic expressivity.

Therefore, we assume that a succinct specification of a set of admissible worlds depending on the respective domain is given. The main reason for this assumption is to simplify the following presentation—we will describe a method to obtain such a set algorithmically in Chapter 4.

Remark 3.2. As already discussed in Section 3.1, for group observations $Obs_{\mathcal{G}}(l)$ every agent $i \in \mathcal{G}$ is aware that all other agents in $\mathcal{G}$ have observed this fact. Together with Definition 3.5, the semantics of observations is then equivalent to the usual semantics of common knowledge. In [FHMV95], a definition of common knowledge is given through the fixed-point axiom: *A fact l is common knowledge among a group $\mathcal{G}$ if and only if all members of $\mathcal{G}$ know that l is true and is common knowledge.* Thus, we could also equivalently use the established common knowledge operator $C_{\mathcal{G}}(l)$ instead of the previously defined observation atoms $Obs_{\mathcal{G}}(l)$. However, the concept of common knowledge is usually used to describe emergent states of agents' knowledge. On the other hand, in the context of our approach, observations are an extrinsic feature that will result in the emergence of other belief states. To keep a clear distinction of the intended use of the operator, we will therefore continue to use $Obs_{\mathcal{G}}(l)$ instead of $C_{\mathcal{G}}(l)$.

Example 3.3 (Trains continued)**.** For Example 3.2, we have ground terms A, B, C_A, C_B, C_C, T_1, and T_2, representing Alice, Bob, three cities, and two trains. Furthermore, we have atoms $on(y, x)$ indicating that person y is on train x, and $at(x, z)$ indicating that train x is at city z. Finally, we have observation atoms of the kind $Obs_{\mathcal{G}}(at(x, z))$, indicating that the agents in $\mathcal{G}$ observe that train x is at station z. A possible world can for example be $\omega_1 = \{at(T_1, C_A), on(A, T_1), Obs_{\{A\}}(at(T_1, C_A))\}$, indicating that train T_1 is at city C_A and A has boarded that train.

We define satisfaction of a ground formula F by a world ω in the usual way [Llo87]:

Definition 3.6 (Satisfaction of ground formulae)**.** Let F, F', F'' be ground formulae and ω a world. Then, F is satisfied by ω (denoted $\omega \models F$) if and only if:

case $F = a$ for some ground atom a:	$a \in \omega$.
case $F = \neg F'$ for some ground formula F':	$\omega \not\models F'$.
case $F = F' \wedge F''$ for formulae F' and F'':	$\omega \models F'$ and $\omega \models F''$.
case $F = F' \vee F''$ for formulae F' and F'':	$\omega \models F'$ or $\omega \models F''$.

We say that a formula F is a tautology if $\omega \models F$ for all admissible worlds $\omega \in \hat{\Omega}$. We say that a formula F is a contradiction if there is no world $\omega \in \hat{\Omega}$ such that $\omega \models F$. We use the usual symbols $\top$ and $\bot$ to denote tautologies and contradictions, respectively.

3.2.2. Threads

To model temporal evolutions of the problem domain we use the definition of *threads* from [SPSS11]:

Definition 3.7 (Thread)**.** A *thread* Th is a mapping from the set of time points τ to the set of admissible worlds: $Th : \tau \rightarrow \hat{\Omega}$

Thus, a thread is a sequence of worlds and $Th(t)$ identifies the actual world at time t according to thread Th. The set of all possible threads (i.e., all possible sequences constructible from τ and $\hat{\Omega}$) is denoted by $\mathcal{T}$. Again, we refrain from directly working with $\mathcal{T}$, and instead assume that any meaningful problem specification gives information about possible temporal evolutions of the system. We use $\hat{\mathcal{T}}$ to represent this set of relevant possible threads. For notational convenience, we assume that there is an additional prior world $Th(0)$ for every thread.

Following Definition 3.6, we use $Th \models F_t$ to denote that thread Th satisfies formulae F at time t (i.e., $Th \models F_t \equiv Th(t) \models F$). Accordingly, we use $\mathcal{T} \models F_t$ to denote that every thread $Th \in \mathcal{T}$ satisfies formula F at time t.

We assume that the system is synchronous, i.e., the agents have a global clock. Thus, even if an agent does not observe anything in world $Th(t)$, it is still aware of time passing and can therefore distinguish between worlds $Th(t)$ and $Th(t-1)$.

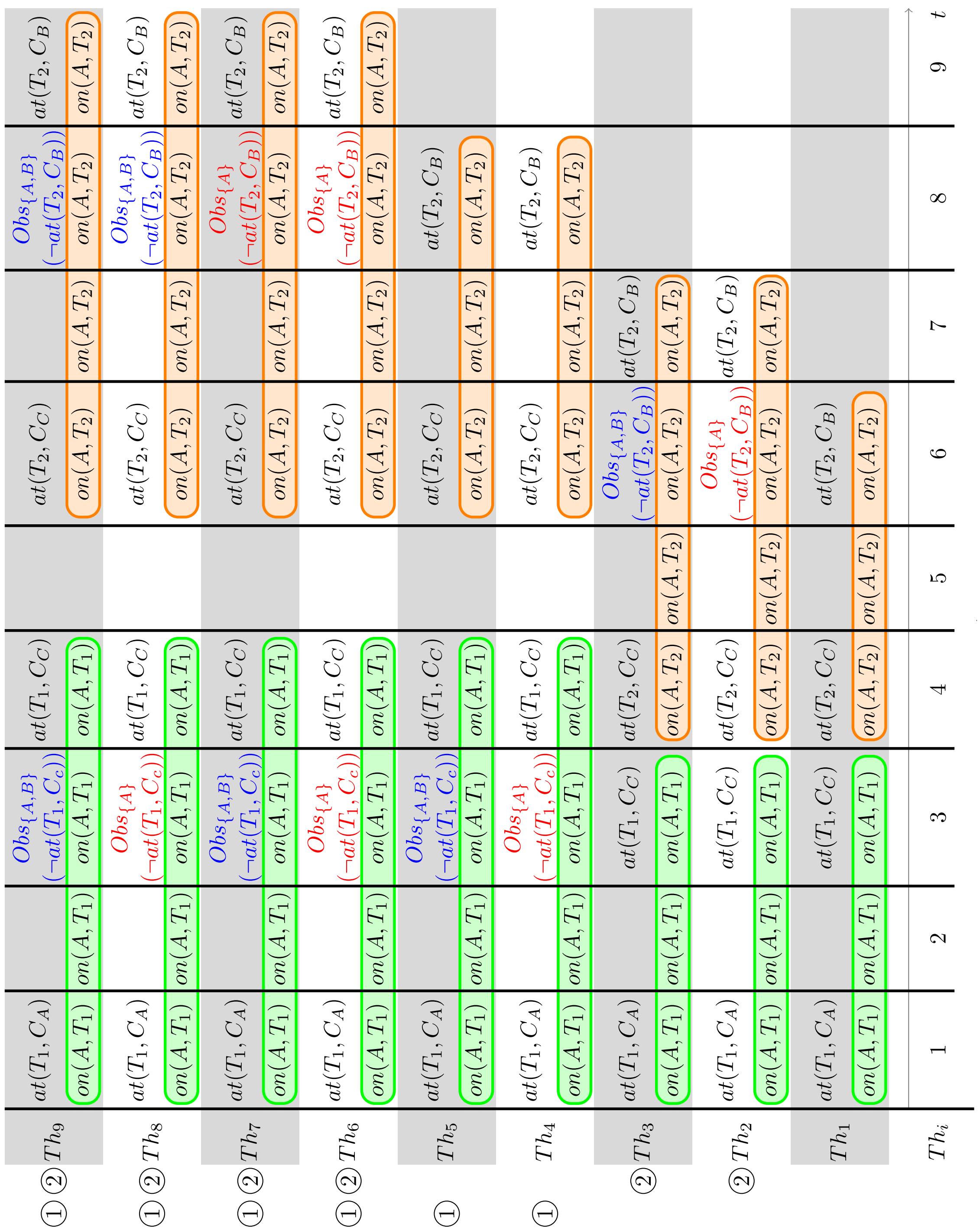
Example 3.4 (Trains continued)**.** The description from Example 3.2 yields the set of possible threads $\hat{\mathcal{T}}$ depicted in Figure 3.1. Note that this is a manually specified set of threads containing only threads that comply with the description in Example 3.2. The set of all possible threads $\mathcal{T}$ would contain a vast number of additional threads that are irrelevant to the described scenario.

3.2.3. Kripke Structures

With the definition of threads, we can use a slightly modified version of Kripke structures [Kri63]. As usual, we define a Kripke structure as a tuple $\langle\hat{\Omega}, \mathcal{K}_1, ..., \mathcal{K}_n\rangle$, with the set of admissible worlds $\hat{\Omega}$ and binary relations $\mathcal{K}_i$ on $\hat{\Omega}$ for every agent $i \in \mathcal{A}$. Thus, the Kripke relation (also called possibility relation) for agent i at world ω is defined as

$$\mathcal{K}_i(\omega) = \{\omega' : (\omega, \omega') \in \mathcal{K}_i\} \tag{3.1}$$

Intuitively, $(\omega, \omega') \in \mathcal{K}_i$ specifies that in world ω, agent i considers ω' also as a possible world. In other words, with its current information agent i is unable to distinguish worlds ω and ω'.



Th_i	1	2	3	4	5	6	7	8	9
① ② Th_9	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$Obs_{\{A,B\}}(\neg at(T_1,C_c))$ $on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$		$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$Obs_{\{A,B\}}(\neg at(T_2,C_B))$ $on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$
① ② Th_8	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$Obs_{\{A\}}(\neg at(T_1,C_c))$ $on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$		$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$Obs_{\{A,B\}}(\neg at(T_2,C_B))$ $on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$
① ② Th_7	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$Obs_{\{A,B\}}(\neg at(T_1,C_c))$ $on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$		$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$Obs_{\{A\}}(\neg at(T_2,C_B))$ $on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$
① ② Th_6	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$Obs_{\{A\}}(\neg at(T_1,C_c))$ $on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$		$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$Obs_{\{A\}}(\neg at(T_2,C_B))$ $on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$
① Th_5	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$Obs_{\{A,B\}}(\neg at(T_1,C_c))$ $on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$		$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$	
① Th_4	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$Obs_{\{A\}}(\neg at(T_1,C_c))$ $on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$		$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$	
② Th_3	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$	$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$Obs_{\{A,B\}}(\neg at(T_2,C_B))$ $on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$		
② Th_2	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$	$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$Obs_{\{A\}}(\neg at(T_2,C_B))$ $on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$		
Th_1	$at(T_1,C_A)$ $on(A,T_1)$	$on(A,T_1)$	$at(T_1,C_C)$ $on(A,T_1)$	$at(T_2,C_C)$ $on(A,T_2)$	$on(A,T_2)$	$at(T_2,C_B)$ $on(A,T_2)$			

Figure 3.1.: Visualization of the possible threads Th_i from Example 3.2. For an easier distinction, shared observations between A and B are marked in blue, single observations of A are marked in red, and all situations where Alice is on train 1 or train 2 are marked in green and orange, respectively. Note that if a train is running late (the respective threads are marked with according circles), there are always two possible threads: one where only A observes this and one where both share the observation.

We initialize the Kripke structure such that all threads are considered possible at time $t = 0$:

$$\forall Th \in \hat{\mathcal{T}} : \ \mathcal{K}_i(Th(0)) = \bigcup_{Th' \in \hat{\mathcal{T}}} \{Th'(0)\}, \ i \in \mathcal{A} \tag{3.2}$$

With the evolution of time, each agent can eliminate the worlds that do not comply with its respective observations. Through the elimination of worlds, an agent will also reduce the set of threads it considers possible (if—due to some observation—a world ω is considered impossible at a time point t, then all threads Th with $Th(t) = \omega$ are considered impossible). We assume that agents have perfect recall and therefore will not consider some thread possible again if it was considered impossible at one point. Thus, $\mathcal{K}_i$ is updated w.r.t. the agent's respective observations, such that it considers all threads possible that both comply with its current observations and were considered possible at the previous time point:

$$\begin{aligned} \mathcal{K}_i(Th(t)) = \big\{Th'(t) : \big(Th'(t-1) \in \mathcal{K}_i(Th(t-1)) \wedge \\ \{Obs_{\mathcal{G}}(l) \in Th(t) : i \in \mathcal{G}\} = \{Obs_{\mathcal{G}}(l) \in Th'(t) : i \in \mathcal{G}\}\big)\big\} \end{aligned} \tag{3.3}$$

The following two corollaries describe key properties of $\mathcal{K}_i$ that follow immediately from the definitions in (3.2) and (3.3):

Corollary 3.1 (Equivalence relation)**.** *$\mathcal{K}_i$ defines an equivalence relation over the possible worlds $\mathcal{K}_i(Th(t))$ for time points $t \in \tau$.*

Corollary 3.2 (Reduction of considered threads)**.** *The set of threads Th' considered possible w.r.t. $\mathcal{K}_i$ is narrowing to a smaller and smaller subset over time, i.e., $\{Th' : Th'(t) \in \mathcal{K}_i(Th(t))\} \subseteq \{Th' : Th'(t-1) \in \mathcal{K}_i(Th(t-1))\}$ for all $Th \in \hat{\mathcal{T}}$ and $t \in \tau$.*

Note that updates of $\mathcal{K}_i$ are defined such that new information is incorporated instantaneously, i.e., if at time t an agent observes some fact, it updates its possibility relations already at time t such that it considers every world impossible that does not comply with the observation of time t.

Example 3.5 (Trains continued)**.** From Figure 3.1, we obtain that at time 1, the only possible world is $\{at(T_1, C_A), on(A, T_1)\}$, which is contained in all possible threads. Thus, $\mathcal{K}_i(Th_j(1))$ contains exactly this world for all agents i and threads j. Consequently, both agents consider all threads as possible at time 1.

Now, assume that time evolves for two steps and the actual thread is Th_4 (i.e., train T_1 is running late, but A does not inform B about this). Both agents will update their possibility relations accordingly, yielding

$$\mathcal{K}_A(Th_4(3)) = \{\{Obs_{\{A\}}(\neg at(T_1, C_C)), on(A, T_1)\}\}$$

and

$$\mathcal{K}_B(Th_4(3)) = \{\{at(T_1, C_C), on(A, T_1)\},\ \{Obs_{\{A\}}(\neg at(T_1, C_C)), on(A, T_1)\}\},$$

i.e., A knows that T_1 is not on time, while B is unaware of T_1 being late, since he still considers a situation possible where train T_1 is at city C_C at time $t = 3$.

3.2.4. Subjective Posterior Temporal Probabilistic Interpretations

Each agent has probabilistic beliefs about the expected evolution over time. This is expressed through subjective temporal probabilistic interpretations:

Definition 3.8 (Subjective posterior probabilistic temporal interpretation)**.** Given a set of possible threads $\hat{\mathcal{T}}$, some thread $\mathring{Th} \in \hat{\mathcal{T}}$, a time point $t' > 0$ and an agent i, the function $\mathcal{I}_{i,t'}^{\mathring{Th}} : \hat{\mathcal{T}} \rightarrow [0, 1]$ specifies the *subjective posterior probabilistic temporal interpretation* from agent i's point of view at time t' in thread $\mathring{Th}$, i.e., a probability distribution over all possible threads: $\sum_{Th \in \hat{\mathcal{T}}} \mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 1$. Since the probabilistic interpretations over possible threads depend on the respective perspective of agent i, $\mathring{Th}$ marks the point of view for a subjective interpretation. Thus, we call $\mathring{Th}$ the *point of view (pov) thread* of interpretation $\mathcal{I}_{i,t'}^{\mathring{Th}}$.

The concept of point of view threads can be seen as conditional probabilities: A subjective posterior probabilistic interpretation $\mathcal{I}_{i,t'}^{\mathring{Th}}$ specifies agent i's probabilistic interpretation at time t' given that $\mathring{Th}$ is the actual thread. Different threads yield different evolutions of the world and—since every possible thread can be taken as a pov thread— may induce different probabilistic interpretations of an agent. Thus, the notion of pov threads allows to reason about hypothetical beliefs of an agent, for instance if possible future beliefs are analyzed or nested beliefs are evaluated.

To simplify notation, we see $\mathcal{I}_{i,t'}^{\mathring{Th}}$ as a vector and occasionally represent a probabilistic interpretation $\mathcal{I}_{i,t'}^{\mathring{Th}}$ over a vector of possible threads $\hat{\mathcal{T}}$ as a vector as well, so that the jth element of $\mathcal{I}_{i,t'}^{\mathring{Th}}$ refers to the probability assigned to thread Th_j.

The prior probabilities of each agent for all threads are then given by $\mathcal{I}_{i,0}^{\mathring{Th}}(Th)$. Since all threads are indistinguishable a priori, there is only a *single* prior distribution needed for each agent (i.e., $\forall Th, \mathring{Th}, \mathring{Th}' \in \hat{\mathcal{T}} : \mathcal{I}_{i,0}^{\mathring{Th}}(Th) = \mathcal{I}_{i,0}^{\mathring{Th}'}(Th)$). Furthermore, in order to be able to reason about nested beliefs (as discussed below), we assume that the prior probability assessments of all agents are commonly known (i.e., all agents know how all other agents assess the prior probabilities of each thread). This in turn requires that all agents have exactly the same prior probability assessment over all possible threads: if two agents have different, but commonly known prior probability assessments, we essentially have an instance of Aumann's well-known problem of "agreeing to disagree" [Aum76]. Intuitively, if differing priors are commonly known, it is common knowledge that (at least) one of the agents is at fault and should revise its probability assessments. As a result, we have only one prior probability distribution which is the same from all viewpoints, denoted by $\mathcal{I}$. Note that $\mathcal{I}$ directly corresponds to the concept of temporal probabilistic interpretations in [SPSS11].

Remark 3.3. We could use the prior probability distribution $\mathcal{I}$ as an alternative method to distinguish between the set of all possible threads $\mathcal{T}$ and the set of threads $\hat{\mathcal{T}}$ relevant to a specific problem domain. To do so, we simply assign all unwanted threads $Th \notin \hat{\mathcal{T}}$ a probability of zero.

Example 3.6 (Trains continued)**.** A meaningful prior interpretation is

$$\mathcal{I}(\hat{\mathcal{T}}) = \begin{pmatrix} 0.7 & 0.02 & 0.09 & 0.02 & 0.09 & 0.01 & 0.02 & 0.02 & 0.03 \end{pmatrix},$$

which assigns the highest probability to Th_1 (no train running late), lower probabilities to the threads where one train is running late and A informs B (Th_3 and Th_5), even lower probabilities to the events that either both trains are running late and A informs B (Th_7, Th_8, and Th_9) or that one train is running late and A does not inform B (Th_2 and Th_4), and lowest probability to the thread where both trains are running late and A does not inform B (Th_6). Note that $\mathcal{I}$ represents the prior interpretation for the train example and thus is the same for every agent $i \in \mathcal{A}$ and every possible pov thread $\mathring{Th}$.

Even though we only have a single prior probability distribution over the set of possible threads, it is still necessary to distinguish the viewpoints of different agents in different threads, as the following definition of interpretation updates shows.

Whenever an agent updates its Kripke relations according to (3.3), it is necessary to update the probabilistic interpretations of that agent to match the new knowledge. An intuitive way to update the probabilities is conditioning on the remaining worlds in the agent's Kripke structure. We want to point out that conditioning is a suitable choice in PDT Logic, although it is known to produce undesired or incorrect results in many cases, most notably in the Monty Hall problem [vS90]. In [GH03] it is discussed that naive conditioning tends to produce errors because updates are carried out in a simplified space where several events are collapsed since they are seemingly one event. If one uses so-called sophisticated conditioning instead (i.e., conditioning in the sophisticated space, which means that all possible events are represented), probabilities are updated correctly. As the semantics of PDT Logic is based on an exhaustive specification of all relevant threads, conditioning in a proper specification of all relevant threads is inherently sophisticated in the sense of [GH03] and will therefore produce correct results. One can easily verify that with the following update rule, well-known probability puzzles such as the Monty Hall Problem [vS90] can be correctly represented in PDT Logic. Thus, we use the following conditioning-based update rule:

Definition 3.9 (Interpretation update)**.** Let i be an agent, t' a time point, and $\mathring{Th}$ a pov thread. Then, if the system is actually in thread $\mathring{Th}$ at time t', agent i's probabilistic interpretation over the set of possible threads is given by the update rule:

$$\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = \begin{cases} \frac{1}{\alpha_{i,t'}^{\mathring{Th}}} \cdot \mathcal{I}_{i,t'-1}^{\mathring{Th}}(Th) & \text{if } Th(t') \in \mathcal{K}_i(\mathring{Th}(t')) \\ 0 & \text{if } Th(t') \notin \mathcal{K}_i(\mathring{Th}(t')) \end{cases} \tag{3.4}$$

with $\frac{1}{\alpha_{i,t'}^{\mathring{Th}}}$ being a normalization factor to ensure that $\sum_{Th \in \hat{\mathcal{T}}} \mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 1$:

$$\alpha_{i,t'}^{\mathring{Th}} = \sum_{\substack{Th \in \hat{\mathcal{T}}, \\ Th(t') \in \mathcal{K}_i(\mathring{Th}(t'))}} \mathcal{I}_{i,t'-1}^{\mathring{Th}}(Th) \tag{3.5}$$

The invocation of $\mathcal{K}_i$ in the update rule yields obvious ramifications about the evolution of interpretations, as stated in the following corollary:

Corollary 3.3 (Nonzero probabilities)**.** *The subjective temporal probabilistic interpretation $\mathcal{I}_{i,t'}^{\mathring{Th}}$ of an agent i assigns nonzero probabilities exactly to the set of threads that i still considers possible at time t', i.e., $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) > 0$ iff $(Th(t), \mathring{Th}(t)) \in \mathcal{K}_i$*

Essentially, the update rule assigns all impossible threads a probability of zero and scales the probabilities of the remaining threads such that they are proportional to the probabilities of the previous time point. With a given prior probability distribution $\mathcal{I}$ over the set of possible threads, the subjective posterior probabilities $\mathcal{I}_{i,t'}^{\mathring{Th}}$ in a specific pov thread $\mathring{Th}$ for all agents i and all time points t' are induced by the respective observations contained in $\mathring{Th}$. We use $\mathcal{I}^{\mathring{Th}}$ to denote the set of all subjective posterior interpretations $\mathcal{I}_{i,t'}^{\mathring{Th}}$ induced in pov thread $\mathring{Th}$.

Example 3.7 (Trains continued)**.** Applying the update rule from (3.4) to the situation described in Example 3.5, with $\mathcal{I}$ as given in Example 3.6, yields the updated interpretation for A:

$$\mathcal{I}_{A,3}^{\mathring{Th}_4} = \begin{pmatrix} 0 & 0 & 0 & 0.4 & 0 & 0.2 & 0 & 0.4 & 0 \end{pmatrix} \tag{3.6}$$

i.e., A considers exactly those threads possible, where the train is running late and she does not inform B (threads Th_4, Th_6, and Th_8). Due to the lack of any new information, B can only eliminate the situations where A does indeed inform

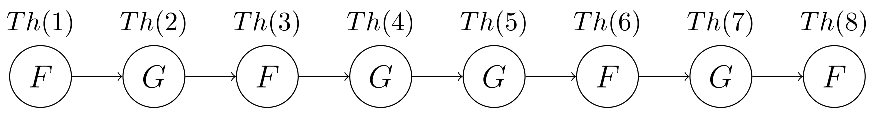


Figure 3.2.: Example thread Th with $\tau = \{1, ..., 8\}$, adopted from [SPSS11]. This figure shows each world that satisfies formula F or formula G.

him about being late at time point 3, and thus B's interpretation is updated to:

$$\mathcal{I}_{B,3}^{\mathring{Th}_4} \approx \begin{pmatrix} 0.82 & 0.02 & 0.10 & 0.02 & 0 & 0.02 & 0 & 0.02 & 0 \end{pmatrix}. \tag{3.7}$$

3.2.5. Frequency Functions

To represent temporal relationships within threads, we adapt the concept of *frequency functions* as introduced in [SPSS11]. Frequency functions provide a flexible way of representing temporal relations between the occurrences of specific events. To illustrate the motivation behind using frequency functions, consider the exemplary thread Th depicted in Figure 3.2. In this thread, one of the events F or G occurs at every time point from $t = 1$ to $t = 8$. As discussed in [SPSS11], there are multiple ways of characterizing temporal relationships between the events F and G: For instance, one might specify how often event F is followed by event G in, say, exactly 2 time points. According to Figure 3.2, this happens in one out of four occurrences of F in Th. It might prove meaningful to exclude the final occurrence of F in Th when determining this frequency, because naturally an occurrence of F at t_{max} cannot be followed by a subsequent occurrence of G. Excluding the final occurrence of F would yield one out of three for the desired frequency. Alternatively, one could also specify how often F is followed by G within the next two time points. For the exemplary thread from Figure 3.2, this would produce frequencies of 1 and 0.75 respectively, again depending on whether the final occurrence of F is included.

This example illustrates already four different possible definitions of temporal relations between events. To maintain flexibility in expressing temporal relations, we do not commit to specific definitions in PDT Logic, but instead we adapt an axiomatic definition of frequency functions:

Definition 3.10 (Frequency functions, adapted from [SPSS11])**.** Let Th be a thread, F and G be ground formulae, and $\Delta t \geq 0$ be an integer. A *frequency function* fr maps quadruples of the form $(Th, F, G, \Delta t)$ to $[0, 1]$ such that the following axioms hold:

(FF1) If G is a tautology, then $\mathsf{fr}(Th, F, G, \Delta t) = 1$.

(FF2) If F is a tautology and G is a contradiction, then $\mathsf{fr}(Th, F, G, \Delta t) = 0$.

(FF3) If F is a contradiction, $\mathsf{fr}(Th, F, G, \Delta t) = 1$.

(FF4) If G is not a tautology, and either F or $\neg G$ is not a tautology, and F is not a contradiction, then there exist threads Th_1, $Th_2 \in \mathcal{T}$ such that $\mathsf{fr}(Th_1, F, G, \Delta t) \neq \mathsf{fr}(Th_2, F, G, \Delta t)$.

Axioms (FF1) to (FF3) ensure that in special cases—i.e., $(G \equiv \top)$, $(F \equiv \bot)$, or $(F \equiv \top,\ G \equiv \bot)$—frequency functions behave as temporal implications with premise F and conclusion G. Axiom (FF4) enforces non-trivial frequency functions by requiring that in all cases not covered by the first three axioms, there must be at least two threads with differing frequency values.

Remark 3.4. This definition mostly corresponds to the definition of frequency functions in [SPSS11], except that we do not require $\Delta t > 0$. In [SPSS11], frequency functions are only intended to express temporal relationships and therefore Δt is limited to nonzero values. By additionally allowing $\Delta t = 0$, we obtain a concise framework that can express both temporal relationships and static constraints within one time point. This will be exploited in the next chapter, where decision procedures for PDT Logic are discussed.

To illustrate the concept of frequency functions, we now present formal definitions for point and existential frequency functions adapted from [SPSS11] that represent the informal descriptions of frequencies from above:

The point frequency function *pfr* expresses how frequently some event F is followed by another event G in *exactly* Δt time units:

$$pfr(Th, F, G, \Delta t) = \frac{|\{t : Th(t) \models F \wedge Th(t + \Delta t) \models G\}|}{|\{t : (t \leq t_{max} - \Delta t) \wedge Th(t) \models F\}|} \tag{3.8}$$

If the denominator is zero, we define *pfr* to be 1. The denominator counts the total number of occurrences of F in a given thread Th and the numerator counts the number of occurrences of F followed by G after exactly Δt time units. Thus, the ratio *pfr* expresses how frequently F is followed by G in exactly Δt time units. Note that the denominator only considers occurrences of F up to time $t_{max} - \Delta t$. This is done to reflect the previously discussed intuition that occurrences of F in the last Δt time points should be excluded from the frequency, because there is no possibility that they can be followed by a subsequent G after Δt time units.

The existential frequency function *efr* expresses how frequently some event F is followed by another event G *within* the next Δt time units:

$$efr(Th, F, G, \Delta t) = \\ \frac{efn(Th, F, G, \Delta t, 0, t_{max})}{|\{t : (t \leq t_{max} - \Delta t) \wedge Th(t) \models F\}| + efn(Th, F, G, \Delta t, t_{max} - \Delta t, t_{max})}, \tag{3.9}$$

with

$$\mathit{efn}(Th, F, G, \Delta t, t_1, t_2) = |\{t : (t_1 < t \leq t_2) \wedge Th(t) \models F \\ \wedge \exists t' \in [t, \min(t_2, t + \Delta t)] \ (Th(t') \models G)\}|$$

The function efn counts the number of occurrences of F followed by a subsequent occurrence of G within the next Δt time units. The first summand of the denominator again counts the total number of occurrences of F up to the time point $t_{max} - \Delta t$. In the second summand of the denominator, additional occurrences of F followed by G within Δt time units. The intuition of this definition is again to exclude occurrences of F in the final Δt time units if they are not followed by G. Since G may occur within the range Δt, but this range cannot be fully considered for the final Δt time points, only occurrences of F with an according subsequent occurrence of G are considered for these final time points. Consequently, the ratio *efr* expresses how frequently some event F is followed by G within the next Δt time units without letting single occurrences of F in the final Δt time points decrease the ratio.

Returning to the exemplary thread Th from Figure 3.2, we can evaluate the frequency functions for the given thread: Suppose that we want to determine how often F is followed by G exactly after two time steps. This can be expressed through a point frequency function:

$$\mathit{pfr}(Th, F, G, 2) = \frac{1}{3}.$$

If instead we want to know how often F is followed by G within the next two time steps, we can use an existential frequency function:

$$\mathit{efr}(Th, F, G, 2) = \frac{3}{3} = 1$$

It should be noted that frequency functions can be used to model temporal relationships usually expressed through temporal operators. For instance, *pfr* with $\Delta t = 1$ reflects the *next* operator and *efr* with $\Delta t = t_{max}$ reflects the *future* operator. The meaning of additional temporal operators such as *until* can be captured through the definition of additional frequency functions, if required.

3.2.6. Semantics of the Belief Operator

Now, with the definitions of subjective posterior probabilistic temporal interpretations and the introduction of frequency functions, we can provide a formal semantics for the belief operators defined in Section 3.1. This semantics extends definitions from [SPSS11] for the satisfiability of static interpretations to obtain a formal definition of probabilistic multi-agent beliefs.

Definition 3.11 (Belief Semantics)**.** Let i be an agent and $\mathcal{I}_{i,t'}^{\mathring{Th}}$ be agent i's interpretation at time t' in pov thread $\mathring{Th}$. Then, it follows from this interpretation that agent i believes at time t' with a probability in the range $[\ell, u]$ that

1. (Belief in ground formulae)
   a formula F holds at time t (denoted by $\mathcal{I}_{i,t'}^{\mathring{Th}} \models B_{i,t'}^{\ell,u}(F_t)$) iff
$$\ell \leq \sum_{Th \in \hat{\mathcal{T}}, Th(t) \models F} \mathcal{I}_{i,t'}^{\mathring{Th}}(Th) \leq u. \tag{3.10}$$

2. (Belief in rules)
   a temporal rule $r_{\Delta t}^{\mathsf{fr}}(F, G)$ holds (denoted by $\mathcal{I}_{i,t'}^{\mathring{Th}} \models B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F, G))$) iff
$$\ell \leq \sum_{Th \in \hat{\mathcal{T}}} \mathcal{I}_{i,t'}^{\mathring{Th}}(Th) \cdot \mathsf{fr}(Th, F, G, \Delta t) \leq u. \tag{3.11}$$

3. (Nested beliefs)
   a belief $B_{j,t}^{\ell_j,u_j}(\varphi)$ of some other agent j holds at time t' (denoted by $\mathcal{I}_{i,t'}^{\mathring{Th}} \models B_{i,t'}^{\ell,u}(B_{j,t}^{\ell_j,u_j}(\varphi))$) iff
$$\ell \leq \sum_{\substack{Th \in \hat{\mathcal{T}} \\ \mathcal{I}_{j,t}^{Th} \models B_{j,t}^{\ell_j,u_j}(\varphi)}} \mathcal{I}_{i,t'}^{\mathring{Th}}(Th) \leq u. \tag{3.12}$$

The intuition behind this semantics is as follows. For beliefs in ground formulae F_t, the subjective posterior probabilities $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th)$ of an agent i at time t' in pov thread $\mathring{Th}$ are added for all threads Th that satisfy F at time t. Thus, the sum in (3.10) represents the exact probability that $\mathcal{I}_{i,t'}^{\mathring{Th}}$ assigns to F_t. If this sum is within the specified boundaries $[\ell, u]$, the respective belief $B_{i,t'}^{\ell,u}(F_t)$ holds for agent i at time t' in pov thread $\mathring{Th}$.

For beliefs in rules, the subjective posterior probabilities $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th)$ for every thread are weighted with the corresponding frequency $\mathsf{fr}(Th, F, G, \Delta t)$ from rule $r_{\Delta t}^{\mathsf{fr}}(F, G)$. Thus, the weighted sum of $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th)$ in (3.11) represents the exact probability that $\mathcal{I}_{i,t'}^{\mathring{Th}}$ assigns to the temporal relation between F and G according to frequency function fr. For beliefs in rules, the belief object $r_{\Delta t}^{\mathsf{fr}}(F, G)$ only contains information about the *type* of frequency function fr, while constraints on the respective frequency *values* are given through the belief quantification $[\ell, u]$, i.e., an agent does not have probabilistic beliefs in specific frequency values.

Remark 3.5. It should be noted that the semantics of beliefs in rules in (3.11) together with the axiomatic definition of frequency functions in Definition 3.10 yields certain constraints on satisfiable beliefs in rules $r^{\mathsf{fr}}_{\Delta t}(F, G)$. If G is a tautology or F is a contradiction (i.e., in Definition 3.10 FF1 or FF3 are satisfied), it holds for the respective frequency function that $\mathsf{fr}(Th, F, G, \Delta t) = 1$ for every possible thread Th, and thus, any belief $B^{\ell,u}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(F, G))$ is satisfiable if and only if the belief is quantified with $u = 1$, regardless of the set of threads $\hat{\mathcal{T}}$ or the corresponding interpretation $\mathring{\mathcal{I}}^{Th}_{i,t'}$. Analogously, if F is a tautology and G is a contradiction (i.e., FF2 is satisfied), any belief $B^{\ell,u}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(F, G))$ is only satisfiable for $\ell = 0$.

For nested beliefs $B^{\ell,u}_{i,t'}(B^{\ell_j,u_j}_{j,t}(\varphi))$, the expression is unnested by first determining all possible pov threads Th for agent j such that $B^{\ell_j,u_j}_{j,t}(\varphi)$ is satisfied. If $B^{\ell_j,u_j}_{j,t}(\varphi)$ corresponds to a belief in a fact or in a rule, (3.10) respectively (3.11) can be used to identify threads Th such that $\mathcal{I}^{Th}_{j,t} \models B^{\ell_j,u_j}_{j,t}(\varphi)$. Otherwise, if φ represents another belief formula, the belief has to be unnested recursively until the innermost belief of the expression is obtained. Then, for all threads Th with $\mathcal{I}^{Th}_{j,t} \models B^{\ell_j,u_j}_{j,t}(\varphi)$, agent i's subjective posterior probabilities $\mathring{\mathcal{I}}^{Th}_{i,t'}(Th)$ are added again to determine whether the outer belief holds. Note that agent i does not know the actual beliefs of agent j. However, due to the assumption of common and equal priors discussed in Section 3.2.4, agent i is able to reason about agent j's hypothetical interpretation updates given that the system is in a specific thread. Thus, agent i is able to compute (3.12) without knowing j's exact beliefs.

Example 3.8 (Trains continued)**.** We can use a point frequency function to express beliefs about the punctuality of trains. Assume that both A and B judge the probability of a train running late (i.e., arriving after 3 instead of 2 time units, expressed through the temporal rule $r^{\mathit{pfr}}_3(at(T_1, C_A), at(T_1, C_C)))$ as being at most 0.4. This yields the following belief formulae

$$\begin{matrix} B^{0,0.4}_{i,0}(r^{\mathit{pfr}}_3(at(T_1, C_A), at(T_1, C_C))) \\ B^{0,0.4}_{i,0}(r^{\mathit{pfr}}_3(at(T_2, C_C), at(T_2, C_B))) \end{matrix}, \quad i \in \{A, B\}. \tag{3.13}$$

For the temporal rules expressed in these belief formulae, we obtain the following frequencies from Figure 3.1:

$$\begin{aligned}
&pfr(Th,\ at(T_1,C_A),\ at(T_1,C_C),\ 3) = 0 && \text{for } Th \in \{Th_1,...,Th_3\}\\
&pfr(Th,\ at(T_1,C_A),\ at(T_1,C_C),\ 3) = 1 && \text{for } Th \in \{Th_4,...,Th_9\}\\
&pfr(Th,\ at(T_2,C_C),\ at(T_1,C_B)\,,3) = 0 && \text{for } Th \in \{Th_1,Th_4,Th_5\}\\
&pfr(Th,\ at(T_2,C_C),\ at(T_1,C_B),\ 3) = 1 && \text{for } Th \in \{Th_2,Th_3,Th_6,...,Th_9\}
\end{aligned}$$

Combining these frequency values with the prior interpretation

$$\mathcal{I}(\hat{\mathcal{T}}) = \begin{pmatrix}0.7 & 0.02 & 0.09 & 0.02 & 0.09 & 0.01 & 0.02 & 0.02 & 0.03\end{pmatrix},$$

given in Example 3.6 yields the sum

$$\sum_{Th\in\hat{\mathcal{T}}} \mathcal{I}(Th)\cdot pfr(Th,\ F,\ G,\ 3) = 0.19$$

for both $F = at(T_1,C_A)$, $G = at(T_1,C_C)$ and $F = at(T_2,C_C), G = at(T_2,C_B)$. As this sum is within the belief quantification $[\ell,u] = [0,\ 0.4]$, the belief formulae in (3.13) are valid. Note that the prior probabilities from Example 3.6 have been specified such that both trains are late with the same probability, and thus the respective sums for the above frequencies are the same.

From the above definitions, we can use the belief about some fact F to quantify the belief about the negation of this fact $\neg F$:

Corollary 3.4 (Belief in negated facts)**.** *Let $B^{\ell,u}_{i,t'}(F_t)$ be an agent's quantified temporal belief about some fact F according to Definition 3.11. Then, the agent's belief in the negation of this fact $\neg F$ is given as $B^{\ell',u'}_{i,t'}(\neg F)$ with $\ell' = 1-u$ and $u' = 1-\ell$.*

3.3. Evolution over Time

In order to completely specify a problem in PDT Logic, we introduce the concept of *doxastic systems.*

Definition 3.12 (Doxastic system)**.** Let $\mathcal{A}$ be a set of agents, $\hat{\mathcal{T}}$ be a set of threads, $A_0^{|\mathcal{A}|\times|\hat{\mathcal{T}}|}$ be a matrix of prior probability distributions across $\hat{\mathcal{T}}$ for every agent in $\mathcal{A}$, and $\mathcal{F}$ be a set of frequency functions. Then, we call the quadruple $\mathcal{D} = \langle \mathcal{A}, \hat{\mathcal{T}}, \mathcal{F}, A_0^{|\mathcal{A}|\times|\hat{\mathcal{T}}|}\rangle$ a *doxastic system.*

Note that several of the parameters discussed before are not explicitly specified in a doxastic system: neither the set of possible worlds Ω, the set of ground atoms $\mathcal{B}$, the set of observation atoms $\mathcal{L}_{obs}$, nor the set of time points τ are explicitly specified. However, all relevant information regarding these parameters is already contained in the specification of $\hat{\mathcal{T}}$.

Remark 3.6. Since all agents share a common prior, all rows of A_0 are the same. Thus, one could obtain a more parsimonious problem specification by only providing the single unique row vector of prior probabilities. The choice of using the matrix A_0 nonetheless is for notational purposes only: it will simplify the presentation of interpretation update operations later on.

Definition 3.13 (Admissibility of doxastic systems)**.** Let $\mathcal{D} = \langle \mathcal{A}, \hat{\mathcal{T}}, \mathcal{F}, A_0^{|\mathcal{A}|\times|\hat{\mathcal{T}}|} \rangle$ be a doxastic system. $\mathcal{D}$ is called admissible iff every world (implicitly) defined in $\hat{\mathcal{T}}$ is admissible (according to Definition 3.5) and all rows of $A_0^{|\mathcal{A}|\times|\hat{\mathcal{T}}|}$ sum to one.

To identify specific situations in a doxastic system after some time has passed and some observations occurred, we furthermore define *pointed doxastic systems*:

Definition 3.14 (Pointed doxastic system, pds)**.** Let $\mathcal{D} = \langle \mathcal{A}, \hat{\mathcal{T}}, \mathcal{F}, A_0^{|\mathcal{A}|\times|\hat{\mathcal{T}}|} \rangle$ be a doxastic system and $\mathcal{H}$ be a set of time-stamped observation atoms such that all observation atoms from $\mathcal{H}$ occur in at least one of the worlds (implicitly) defined in $\hat{\mathcal{T}}$. Then we call the pair $\langle \mathcal{D}, \mathcal{H} \rangle$ a *pointed doxastic system.*

Definition 3.15 (Admissibility of pointed doxastic systems)**.** Let $\langle \mathcal{D}, \mathcal{H} \rangle$ be a pointed doxastic system, and $\hat{\mathcal{T}}$ the set of threads from $\mathcal{D}$. $\langle \mathcal{D}, \mathcal{H} \rangle$ is called admissible iff $\mathcal{D}$ is admissible and there exists a thread $Th \in \hat{\mathcal{T}}$ such that $\forall Obs_{\mathcal{G}}(l)_t \in \mathcal{H} : Obs_{\mathcal{G}}(l) \in Th(t)$ (i.e., $\hat{\mathcal{T}}$ must contain at least one thread that complies with all timed observations from $\mathcal{H}$).

Intuitively, the set of timed observations specified in a pds points to a certain situation in a doxastic system. One could view $\hat{t}(\mathcal{H}) = \max\{t : \exists Obs_{\mathcal{G}}(l)_t \in \mathcal{H}\}$ as the present time in a pds: the most recent observation occurred at $\hat{t}(\mathcal{H})$, all observations that actually occurred in the past ($t < \hat{t}(\mathcal{H})$) are specified in $\mathcal{H}$ (and are thus deterministic in retrospective), and no further information about future observations $t > \hat{t}(\mathcal{H})$ is given. In this sense, $\mathcal{H}$ specifies a certain history up to $\hat{t}(\mathcal{H})$ in a doxastic system and points to the last event of this history.

Example 3.9 (Trains continued)**.** A doxastic system for the train example can be specified as

$$\mathcal{D} = \langle \{A, B\}, \{Th_1, ..., Th_9\}, \{pfr, efr\}, A_0 \rangle,$$

with

$$A_0 = \begin{pmatrix} 0.7 & 0.02 & 0.09 & 0.02 & 0.09 & 0.01 & 0.02 & 0.02 & 0.03 \\ 0.7 & 0.02 & 0.09 & 0.02 & 0.09 & 0.01 & 0.02 & 0.02 & 0.03 \end{pmatrix}.$$

To identify the situation described in Example 3.5 (T_1 is running late), we can specify the following pointed doxastic system:

$$\langle \mathcal{D}, \{Obs_{\{A\}}(\neg at(T_1, C_C)_3)\}\rangle$$

3.3.1. Evolution of Probabilistic Interpretations

In accordance with the prior probability matrix A_0 from Definition 3.12, we define an interpretation matrix $A_t^{\mathring{Th}}$ to store the interpretations of all agents $\mathcal{A}$ (with n denoting the number of agents $|\mathcal{A}|$) across all threads $Th_1, ..., Th_m$ given that the doxastic system is in pov thread $\mathring{Th}$ at time t:

$$A_t^{\mathring{Th}} = \begin{pmatrix} \mathcal{I}_{1,t}^{\mathring{Th}}(Th_1) & \dots & \mathcal{I}_{1,t}^{\mathring{Th}}(Th_m) \\ \vdots & \ddots & \vdots \\ \mathcal{I}_{n,t}^{\mathring{Th}}(Th_1) & \dots & \mathcal{I}_{n,t}^{\mathring{Th}}(Th_m) \end{pmatrix} \tag{3.14}$$

With the definition of $\mathcal{K}_i$ from (3.3), the update rule from (3.4), and using the prior probability matrix A_0 from Definition 3.12, we can provide an update matrix $U_t^{\mathring{Th}}$ to calculate the interpretation matrix for any pov thread $\mathring{Th}$ at any time point t ($\circ$ denotes the element-wise multiplication of matrices):

$$A_t^{\mathring{Th}} = A_{t-1}^{\mathring{Th}} \circ U_t^{\mathring{Th}} \tag{3.15}$$

with

$$(u_t^{\mathring{Th}})_{ij} = \begin{cases} 0 & \text{if } Th_j(t) \notin \mathcal{K}_i(\mathring{Th}(t)) \\ \frac{1}{\alpha_{i,t'}^{\mathring{Th}}} & \text{if } Th_j(t) \in \mathcal{K}_i(\mathring{Th}(t)) \end{cases} \tag{3.16}$$

and $\alpha_{i,t'}^{\mathring{Th}}$ a normalization factor as defined in (3.5).

The time-stamped observations specified in the history $\mathcal{H}$ of a pds $\langle \mathcal{D}, \mathcal{H}\rangle$ induce an *updated set of reachability relations* $\mathcal{K}_i(Th(t))$ for every thread Th that complies with the given observations (for threads $Th_\perp$ that do not comply with the given observations $\mathcal{K}_i(Th_\perp(t)) = \emptyset$). These updated reachability relations in turn yield the updated interpretations in $A_t^{\mathring{Th}}$. The complete state of interpretations at any time point for every possible pov thread $\mathring{Th}_1, ..., \mathring{Th}_m$ can then be specified as a block matrix, which we call the *belief state* (bs) of a pds at time t:

$$bs(\langle \mathcal{D}, \mathcal{H}\rangle, t) = \begin{pmatrix} A_t^{\mathring{Th}_1}, & ..., & A_t^{\mathring{Th}_m} \end{pmatrix} \tag{3.17}$$

We use $bs(\langle \mathcal{D}, \mathcal{H} \rangle)$ to denote the sequence of all belief states $bs(\langle \mathcal{D}, \mathcal{H} \rangle, t)$ from $t = 1$ to $t = t_{max}$.

This definition of belief states can be seen as a specification of conditional probabilities: the kth entry of $bs(\langle \mathcal{D}, \mathcal{H} \rangle, t)$ specifies the interpretations of all agents across all threads at time t given that the system is in pov thread $\mathring{Th}_k$. Thus—as every thread is considered as a potential pov thread—a full specification of an agent's belief state for m threads requires $m \cdot m$ conditional probabilities for every time point t. This is a very general representation of belief states to allow for an easy evaluation of subjective posterior interpretations at arbitrary time points and pov threads and for an intuitive definition of belief state updates. However, this general definition contains some redundant information. By leveraging certain properties of the semantics of PDT Logic, we identify means to obtain compressed representations of the belief state in the following.

Corollary 3.5 (Null vectors in $A_t^{\mathring{Th}_k}$)**.** *Due to the definition of (3.16), the ith row of $A_t^{\mathring{Th}_k}$ is $\vec{0}$ iff agent i's actual observations (as specified in $\mathcal{H}$) do not match the observations specified in thread Th_k.*

Proposition 3.6 (Belief state compression)**.** *Let $\langle \mathcal{D}, \mathcal{H} \rangle$ be a pointed doxastic system and let t be a time point such that $t \leq \hat{t}(\mathcal{H})$. Then, without any loss of information, the belief state $bs(\langle \mathcal{D}, \mathcal{H} \rangle, t)$ at time t can be represented through*

$$bs(\langle \mathcal{D}, \mathcal{H} \rangle, t)' = \left(\vec{v}_{1,t}, \ ..., \ \vec{v}_{n,t}\right)^T \tag{3.18}$$

with one probability distribution vector $\vec{v}_{i,t}$ per agent i.

Proof. It follows directly from Corollaries 3.3 and 3.5 that the matrices $A_t^{\mathring{Th}_k}$ from $bs(\langle \mathcal{D}, \mathcal{H} \rangle, t)$ with nonzero rows i are exactly those that correspond to threads considered possible by agent i at time t.

From the properties of $\mathcal{K}_i$ given in Corollary 3.1 it follows that all worlds $Th'(t) \in \mathcal{K}_i$ for $t \leq \hat{t}(\mathcal{H})$ are indistinguishable to agent i and therefore are associated with the same interpretation. Thus, all nonzero ith rows of the matrices in bs are identical. Defining $\vec{v}_{i,t}$ as these unique nonzero rows i of bs, we obtain the representation of (3.18). Information about impossible pov threads (as described in Corollary 3.5) is still maintained as they are assigned a probability of 0 in $\vec{v}_{i,t}$. □

It is important to note that this compressed representation is only applicable to time points $t \leq \hat{t}(\mathcal{H})$, because in retrospective an agent is able to classify threads into two categories: those that comply with the observations so far (i.e., those that are considered possible), and those that do not. For time points $t > \hat{t}(\mathcal{H})$ this classification is not possible because $\mathcal{K}_i(Th(t))$ then depends on future observations and can therefore lead to a branching of several distinct interpretations depending on the respective observations.

3.3.2. Evolution of Beliefs

In order to analyze the temporal evolution of beliefs, we use the update rule from (3.15) to update belief states. Since different possible observations yield different branches in the evolution of beliefs, we have to update every thread in the belief state individually, using the respective update matrices $U_t^{\mathring{Th}}$ as defined in (3.16):

$$bs(\langle\mathcal{D},\mathcal{H}\rangle,t) = bs(\langle\mathcal{D},\mathcal{H}\rangle,t-1) \circ (U_t^{\mathring{Th}_1},...,U_t^{\mathring{Th}_m}) \tag{3.19}$$

Furthermore, to analyze satisfiability and validity of arbitrary finite belief expressions $B_{i,t'}^{\ell,u}(\varphi)$ w.r.t. a given pds $\langle\mathcal{D},\mathcal{H}\rangle$, we define an auxiliary belief vector $\vec{b}(\varphi)$ for different beliefs $B_{i,t'}^{\ell,u}(\varphi)$. This vector $\vec{b}(\varphi)$ contains one entry $(\vec{b}(\varphi))_j$ for every possible thread $Th_j \in \hat{\mathcal{T}}$ and is defined as follows:

$$\begin{aligned}
&a) \quad B_{i,t'}^{\ell,u}(F_t): && (\vec{b}(F_t))_j = \begin{cases} 1 \text{ if } Th_j(t) \models F \\ 0 \text{ if } Th_j(t) \not\models F \end{cases}\\
&b) \quad B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F,G)): && (\vec{b}(r_{\Delta t}^{\mathsf{fr}}(F,G)))_j = \mathsf{fr}(Th_j,F,G,\Delta t)\\
&c) \quad B_{i,t'}^{\ell,u}(B_{k,t}^{\ell_k,u_k}(\varphi)): && (\vec{b}(B_{k,t}^{\ell_k,u_k}(\varphi)))_j = \begin{cases} 1 \text{ if } \mathcal{I}_{k,t}^{Th_j} \models B_{k,t}^{\ell_k,u_k}(\varphi) \\ 0 \text{ if } \mathcal{I}_{k,t}^{Th_j} \not\models B_{k,t}^{\ell_k,u_k}(\varphi) \end{cases}
\end{aligned} \tag{3.20}$$

Note that in the case of nested beliefs, the respective entries $(\vec{b}(\varphi))_j$ are set to one if the inner belief holds in thread Th_j, i.e., it is assumed that Th_j is the point of view thread for agent k and then it is checked whether k's belief $B_{k,t}^{\ell_k,u_k}(\varphi)$ is satisfied in this thread.

Using (3.19) and (3.20), we can determine a matrix $P_{t'}(\varphi)$ with the probabilities $p_{i,t'}^{\mathring{Th}_k}(\varphi)$ that each agent i assigns at time t' to some event φ, for all possible pov threads $\mathring{Th}_1,...,\mathring{Th}_m$:[3]

$$P_{t'}(\varphi) = bs(\langle\mathcal{D},\mathcal{H}\rangle,t') \cdot \left(\vec{b}(\varphi),...,\vec{b}(\varphi)\right)^T = \begin{pmatrix} p_{1,t'}^{\mathring{Th}_1} & \cdots & p_{1,t'}^{\mathring{Th}_m} \\ \vdots & \ddots & \vdots \\ p_{n,t'}^{\mathring{Th}_1} & \cdots & p_{n,t'}^{\mathring{Th}_m} \end{pmatrix}(\varphi) \tag{3.21}$$

For n agents and m threads, this results in a $n \times m$ matrix. The rows of this matrix can be seen as conditional probabilities: agent i believes at time t' that a fact φ is true with probability $p_{i,t'}^{\mathring{Th}_k}(\varphi)$ given that the system is in pov thread $\mathring{Th}_k$.

[3]Since we have to consider every possible pov thread $\mathring{Th}_k$, we have to multiply every matrix $A_t^{\mathring{Th}} \in bs(\langle\mathcal{D},\mathcal{H}\rangle,t)$ with $\vec{b}(\varphi)$, thus we need to use the vector $\left(\vec{b}(\varphi),...,\vec{b}(\varphi)\right)^T$ with m rows.

Remark 3.7. Computation of $P_{t'}(\varphi)$ is straightforward for cases 3.20.a) and 3.20.b). To compute the probabilities for nested beliefs in 3.20.c), we start with computing the innermost belief (which is an instance of case 3.20.a) or case 3.20.b) since we assume finite expressions), and then compute the nested beliefs iteratively.

Using Definition 3.11 and Equation (3.21), we can provide a definition for the satisfiability and validity of beliefs:

Definition 3.16 (Validity and satisfiability of beliefs)**.** Let $\mathscr{B}$ be a belief formula as defined in Definition 3.4, $\langle \mathcal{D}, \mathcal{H} \rangle$ a pointed doxastic system, and $P_{t'}(\varphi)$ the corresponding matrix of probabilities at time t' as defined in (3.21). $\mathscr{B}$ is satisfiable (valid) w.r.t. $\langle \mathcal{D}, \mathcal{H} \rangle$ iff

1. For $\mathscr{B} = B^{\ell,u}_{i,t'}(\varphi)$:
   For at least one (all) thread(s) $\mathring{Th}_k \in \hat{\mathcal{T}}$, the entries in row i of $P_{t'}(\varphi)$ satisfy $\ell \leq p^{\mathring{Th}_k}_{i,t'}(\varphi)$ and $u \geq p^{\mathring{Th}_k}_{i,t'}(\varphi)$.
2. For $\mathscr{B} = \neg B^{\ell,u}_{i,t'}(\varphi)$:
   For at least one (all) thread(s) $\mathring{Th}_k \in \hat{\mathcal{T}}$, the entries in row i of $P_{t'}(\varphi)$ satisfy $\ell > p^{\mathring{Th}_k}_{i,t'}(\varphi)$ or $u < p^{\mathring{Th}_k}_{i,t'}(\varphi)$.
3. For $\mathscr{B} = \mathscr{B}_1 \wedge \mathscr{B}_2$:
   For at least one (all) thread(s) $\mathring{Th}_k \in \hat{\mathcal{T}}$, the entries in the corresponding rows of $P_{t'}(\varphi)$ satisfy both $\mathscr{B}_1$ and $\mathscr{B}_2$.
4. For $\mathscr{B} = \mathscr{B}_1 \vee \mathscr{B}_2$:
   $\mathscr{B}_1$ is satisfiable (valid) or $\mathscr{B}_2$ is satisfiable (valid).

Remark 3.8. The distinction between valid and satisfiable belief formulae is only of interest for beliefs at time $t > \hat{t}(\mathcal{H})$. For time points $t \leq \hat{t}(\mathcal{H})$ an agent's belief is uniquely determined through the given observations (cf. Proposition 3.6), resulting in a single probability associated to any belief. Therefore, all invalid belief formulae for $t \leq \hat{t}(\mathcal{H})$ are unsatisfiable.

From Definition 3.4 it follows that the belief object of an atomic belief formula $\mathscr{B}$ as in Definition 3.16-1 can again be any arbitrary belief formula. If the inner belief formula $\mathscr{B}'$ is one of the cases defined in Definition 3.16, validity and satisfiability of the entire expression $\mathscr{B} = B^{\ell,u}_{i,t'}(\mathscr{B}')$ follows inductively from the above definition: If for at least one (all) thread(s) $\mathring{Th}_k \in \hat{\mathcal{T}}$, both the inner belief formula $\mathscr{B}'$ is satisfied and the limits for the outer belief of the respective thread are satisfied, the entire belief formula is $\mathscr{B} = B^{\ell,u}_{i,t'}(\mathscr{B}')$ satisfiable (valid).

Definition 3.16 gives rise to an important property of the belief operator, as the following lemma shows:

Lemma 3.7 (Distributivity of the belief operator). *Let $\mathscr{B} = B^{\ell,u}_{i,t'}(\varphi_1 \otimes \varphi_2)$ be a belief formula with a belief object $(\varphi_1 \otimes \varphi_2)$ and a connective $\otimes \in \{\wedge, \vee\}$. Then, we can express $\mathscr{B}$ equivalently as $\mathscr{B}' = B^{\ell,u}_{i,t'}(\varphi_1) \otimes B^{\ell,u}_{i,t'}(\varphi_2)$.*

Proof. This result follows immediately from the validity and satisfiability of beliefs in Definition 3.16:

The formula $\mathscr{B} = B^{\ell,u}_{i,t'}(\varphi_1 \vee \varphi_2)$ is satisfiable (valid) iff for at least one (all) thread(s) $\mathring{Th}_k \in \hat{\mathcal{T}}$ it holds that $\mathring{Th}_k \models \varphi_1$ or $\mathring{Th}_k \models \varphi_2$ and the respective entries in $P_{t'}(\varphi)$ satisfy Definition 3.16-1. For the former case, $B^{\ell,u}_{i,t'}(\varphi_1)$ is satisfiable (valid) as well, while for the latter case $B^{\ell,u}_{i,t'}(\varphi_2)$ is satisfiable (valid), which reflects exactly the definition of disjunctive belief formulae from Definition 3.16-4. Thus, $\mathscr{B}' = B^{\ell,u}_{i,t'}(\varphi_1) \vee B^{\ell,u}_{i,t'}(\varphi_2)$ is satisfiable (valid) iff $\mathscr{B} = B^{\ell,u}_{i,t'}(\varphi_1 \vee \varphi_2)$ is satisfiable (valid).

Similarly, the formula $\mathscr{B} = B^{\ell,u}_{i,t'}(\varphi_1 \wedge \varphi_2)$ is satisfiable (valid) iff for at least one (all) thread(s) $\mathring{Th}_k \in \hat{\mathcal{T}}$ it holds that both $\mathring{Th}_k \models \varphi_1$ and $\mathring{Th}_k \models \varphi_2$ hold and the respective entries in $P_{t'}(\varphi)$ satisfy Definition 3.16-1. Then, both $B^{\ell,u}_{i,t'}(\varphi_1)$ and $B^{\ell,u}_{i,t'}(\varphi_2)$ are satisfiable (valid) and thus, the formula $\mathscr{B}' = B^{\ell,u}_{i,t'}(\varphi_1) \wedge B^{\ell,u}_{i,t'}(\varphi_2)$ is satisfiable (valid) according to definition Definition 3.16-3. Thus, $\mathscr{B}' = B^{\ell,u}_{i,t'}(\varphi_1) \wedge B^{\ell,u}_{i,t'}(\varphi_2)$ is satisfiable (valid) iff $\mathscr{B} = B^{\ell,u}_{i,t'}(\varphi_1 \wedge \varphi_2)$ is satisfiable (valid). □

To illustrate the evolution of beliefs, we finish the train example with an analysis of expected arrival times.

Example 3.10 (Trains continued). From $\mathcal{D}$, as specified in Example 3.9, we can infer that Bob (and of course Alice, too) can safely assume at time 1 that Alice will arrive at time 8 at the latest with a probability in the range $[0.9, 1]$, as expressed in the belief formula

$$\mathscr{B}_{B,t} = B^{0.9,1}_{B,t}(r^{efr}_7(on(A,T_1), (at(T_2,C_B) \wedge on(A,T_2)))) \tag{3.22}$$

with $t = 1$. For this rule, we obtain the frequencies

$$\begin{aligned} &\mathit{efr}(Th,\ at(T_1,C_A),\ (at(T_2,C_B) \wedge on(A,T_2)),\ 7) = 1 \quad \text{for } Th \in \{Th_1, ..., Th_5\},\\ &\mathit{efr}(Th,\ at(T_1,C_A),\ (at(T_2,C_B) \wedge on(A,T_2)),\ 7) = 0 \quad \text{for } Th \in \{Th_6, ..., Th_9\}, \end{aligned}$$

i.e., in threads $Th_1, ..., Th_5$ from Figure 3.1, the event $(at(T_2,C_B) \wedge on(A,T_2))$ occurs within 7 time points following the event $on(A,T_1)$ from time $t = 1$ (and thus at time $t = 8$ at latest), while in threads $Th_6, ..., Th_9$, the event $(at(T_2,C_B) \wedge$

$on(A, T_2))$ occurs only at time $t = 9$, which is outside of the scope of r_7^{efr} and thus yields a frequency of zero.

At time point 1, Bob still considers all threads as possible, and thus Bob's subjective posterior probabilistic interpretation

$$\mathcal{I}_{B,1}^{\mathring{Th}}(\hat{\mathcal{T}}) = \begin{pmatrix} 0.7 & 0.02 & 0.09 & 0.02 & 0.09 & 0.01 & 0.02 & 0.02 & 0.03 \end{pmatrix}$$

is equal to the prior interpretation given in Example 3.6 for all possible pov threads $\mathring{Th}$. Combining this interpretation with the frequencies given above yields the sum

$$\sum_{Th \in \hat{\mathcal{T}}} \mathcal{I}_{B,1}^{\mathring{Th}}(Th) \cdot \mathit{efr}(Th,\ at(T_1, C_A),\ (at(T_2, C_B) \wedge on(A, T_2)),\ 7) = 0.92$$

and thus formula $\mathscr{B}_{B,1}$ is valid.

Now, consider the previously described situation, where T_1 is running late and A does not inform B about it. This leads to the updated interpretations given in (3.6) and (3.7), i.e.,

$$\begin{aligned} \mathcal{I}_{A,3}^{\mathring{Th}_4} &= (\quad 0 \quad 0 \quad 0 \quad 0.4 \quad 0 \quad 0.2 \quad 0 \quad 0.4 \quad 0\ ), \quad \text{and} \\ \mathcal{I}_{B,3}^{\mathring{Th}_4} &\approx (\ 0.82 \ \ 0.02 \ \ 0.10 \ \ 0.02 \ \ 0 \ \ 0.02 \ \ 0 \ \ 0.02 \ \ 0\ ). \end{aligned}$$

These updates lead to a significant divergence in the belief of the expected arrival time: The corresponding sum with respect to Alice's updated interpretation is

$$\sum_{Th \in \hat{\mathcal{T}}} \mathcal{I}_{A,3}^{\mathring{Th}_4}(Th) \cdot \mathit{efr}(Th,\ at(T_1, C_A),\ (at(T_2, C_B) \wedge on(A, T_2)),\ 7) = 0.4, \tag{3.23}$$

$$\tag{3.24}$$

obtained by Alice's subjective posterior probability assignment of thread Th_4, which is the only nonzero summand in the above sum; all other threads Th are either impossible from Alice's point of view (i.e., $\mathcal{I}_{A,3}^{\mathring{Th}_4}(Th) = 0$ for threads $Th \in \{Th_1, Th_2, Th_3, Th_5, Th_7, Th_9\}$), or the corresponding frequency is zero (for threads Th_6 and Th_8). Thus, Alice's belief in arriving at time point 8 at the latest is drastically reduced, as the lower bound ℓ of Alice's belief may not exceed 0.4. For instance,

$$B_{A,3}^{0.4,1}\big(r_8^{\mathit{efr}}(on(A, T_1), (at(T_2, C_B) \wedge on(A, T_2)))\big), \tag{3.25}$$

is now a valid belief formula. The corresponding sum for Bob's belief at time point 3 is

$$\sum_{Th \in \hat{\mathcal{T}}} \mathcal{I}_{B,3}^{Th_4}(Th) \cdot \mathit{efr}(Th,\, at(T_1, C_A),\, (at(T_2, C_B) \wedge on(A, T_2)),\, 7) = 0.96, \quad (3.26)$$

obtained by summing over Bob's subjective posterior interpretations for threads $Th_1, ..., Th_4$; the remaining threads again only contribute zero summands because either Bob's probability assignment or the corresponding frequency is zero for those threads. Thus, Bob's previous belief (expressed in (3.22)) remains valid at time point $t = 3$, denoted by $\mathscr{B}_{B,3}$.

Even though Alice's beliefs have changed significantly, she is aware that Bob maintains beliefs conflicting with her own, as is shown by the following valid expression of nested beliefs:

$$B_{A,3}^{1,1}(\mathscr{B}_{B,3})$$

To verify that this nested belief holds, we need to consider all threads that Alice considers possible (Th_4, Th_6, Th_8) and determine what Bobs hypothetical beliefs would be in these threads. For Th_4, this has already been analyzed in (3.26). Since threads Th_4, Th_6, and Th_8 are indistinguishable to Bob at time point 3, the same analysis results hold for all three threads. Consequently, $\mathscr{B}_{B_3}$ holds in every thread that Alice considers possible and therefore the sum for this nested belief is

$$\sum_{\substack{Th \in \hat{\mathcal{T}} \\ \mathcal{I}_{B,3}^{Th} \models \mathscr{B}_{B,3}}} \mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 1,$$

i.e., Alice *knows* that Bob's belief is outdated.

Finally, consider the pointed doxastic system $\langle \mathcal{D}, Obs_{\{AB\}}(\neg at(T_1, C_C))_3 \rangle$, i.e., the same situation as before with the only difference that Alice now shares her observation of the delayed train with Bob. It immediately follows that Bob updates his beliefs in the same way as Alice, which in turn yields an update in Alice's beliefs about Bob's beliefs so that now the following expression is valid (because 1 is not a valid lower bound any longer):

$$\neg B_{A,3}^{1,1}(\mathscr{B}_{B,3})$$

This example shows how Alice can reason about the influence of her own actions on Bob's belief state and therefore she can decide on actions that improve Bob's utility (as he does not have to wait in vain). How to find such actions that cause a belief formulae to be valid is another problem that we will treat formally in Chapter 6.

3.4. Concluding Remarks

By extending APT Logic to temporal scenarios with multiple agents, we have developed a general framework to represent and reason about the belief change in multi-agent systems. Next to lifting the single-agent case of APT Logic to multiple agents, we have also provided a suitable semantics to the temporal evolution of beliefs. The resulting framework extends previous work on dynamic multi-agent epistemic logics by enabling the quantification of agents' beliefs through probability intervals and thereby facilitating the use of imprecise probabilities. An explicit notion of temporal relationships is provided through temporal rules building on the concept of frequency functions.

In this chapter we have specified the formal syntax and semantics of PDT Logic and we have shown through examples how this formalism can be used to model agents' beliefs and analyze their respective updates after receiving new information. The formalism introduced in this chapter provides the formal foundation for the following chapters.

Chapter 4

Satisfiability Checking for PDT Logic

In this chapter we will describe procedures to check whether there exists a model for some given set of belief formulae $\mathfrak{B}$. We start with formally defining the satisfiability checking problem in PDT Logic. Using the semantics from the previous chapter, we derive a model checking algorithm based on fully specified doxastic systems. Afterwards, we show how a set of belief formulae can be used to specify a problem in PDT Logic and—together with a given set of threads—how this can be transformed into a mixed integer linear program in order to employ existing solvers to decide satisfiability of PDT Logic formulae. Finally, we show how suitable threads can be derived from a given set of belief formulae automatically.

If a fully specified doxastic system $\langle \mathcal{D}, \mathcal{H} \rangle$ is given, we can define the problem of checking whether a set of belief formulae $\mathfrak{B}$ is satisfiable with respect to this doxastic system as follows. Recall from Section 3.2.4 that we use $\mathcal{I}^{\mathring{Th}}$ to denote the set of all subjective posterior interpretations $\mathcal{I}_{i,t'}^{\mathring{Th}}$ induced by a prior interpretation $\mathcal{I}$ in a pov thread $\mathring{Th}$.

Definition 4.1 (Satisfiability Checking for PDT Logic)**.** Let $\langle \mathcal{D}, \mathcal{H} \rangle$ be a pointed doxastic system with the set of threads $\hat{\mathcal{T}}$ and according prior interpretation $\mathcal{I}$ specified in $\langle \mathcal{D}, \mathcal{H} \rangle$, and $\mathfrak{B}$ be a set of belief formulae. We say that $\mathfrak{B}$ is *satisfiable* w.r.t. $\langle \mathcal{D}, \mathcal{H} \rangle$ if there exists a thread $\mathring{Th}$ in $\hat{\mathcal{T}}$ such that the corresponding interpretations satisfy all belief formulae $\mathscr{B}$ from $\mathfrak{B}$:

$$sat(\mathfrak{B}, \langle \mathcal{D}, \mathcal{H} \rangle) \equiv \exists \mathring{Th} \in \hat{\mathcal{T}} : \left(\forall \mathscr{B} \in \mathfrak{B} : \mathcal{I}^{\mathring{Th}} \models \mathscr{B} \right) \tag{4.1}$$

If such a specification is given, checking satisfiability of $\mathfrak{B}$ with respect to $\langle \mathcal{D}, \mathcal{H} \rangle$ corresponds to checking whether $\langle \mathcal{D}, \mathcal{H} \rangle$ is a model for $\mathfrak{B}$. We continue with introducing a model checking procedure for this fully specified input. Afterwards, we discuss how satisfiability of a set of belief formulae $\mathfrak{B}$ can be decided if no prior probabilities, or neither threads nor prior probabilities are given.

Algorithm 1 Model Checking

```
procedure ModelChecking($(\langle\mathcal{D},\mathcal{H}\rangle, \mathfrak{B})$)
    $bs(\langle\mathcal{D},\mathcal{H}\rangle, 0) \leftarrow (A_0^{Th_1}, ..., A_0^{Th_m})$
    for $t \leftarrow 1, t_{max}$ do                                    ▷ compute all belief states
        $bs(\langle\mathcal{D},\mathcal{H}\rangle, t) \leftarrow bs(\langle\mathcal{D},\mathcal{H}\rangle, t-1) \circ (U_t^{Th_1}, ..., U_t^{Th_m})$
    for $\mathscr{B} \in \mathfrak{B}$ do
        for $\mathring{Th}_k \in \hat{\mathcal{T}}$ do
            if not Check($bs(\langle\mathcal{D},\mathcal{H}\rangle), \mathring{Th}_k, \mathscr{B}$) then    ▷ check if $\mathscr{B}$ is satisfied in $\mathring{Th}_k$
                $\hat{\mathcal{T}} \leftarrow \hat{\mathcal{T}} \setminus \{\mathring{Th}_k\}$    ▷ otherwise remove $\mathring{Th}_k$ from threads to check
                if $\hat{\mathcal{T}} = \emptyset$ then
                    return false                                   ▷ exit if no $\mathring{Th}$ can satisfy $\mathfrak{B}$
    return true                       ▷ success if $\hat{\mathcal{T}}$ is nonempty after checking all $\mathscr{B} \in \mathfrak{B}$

function Check($bs(\langle\mathcal{D},\mathcal{H}\rangle), \mathring{Th}_k, \mathscr{B}$)
    switch ($\mathscr{B}$)                                    ▷ check formulae according to Def. 3.16
        case $B_{i,t'}^{\ell,u}(\varphi)$:
            if $\varphi = \mathscr{B}'$ then    ▷ check nested belief formulae recursively ($\mathscr{B}'$ is a belief formula)
                if not Check($bs(\langle\mathcal{D},\mathcal{H}\rangle), \mathring{Th}_k, \mathscr{B}'$) then
                    return false
            $P_{t'} \leftarrow bs(\langle\mathcal{D},\mathcal{H}\rangle, t') \cdot \vec{b}(\varphi)$    ▷ use $\vec{b}(\varphi)$ from (3.20) to compute $P_{t'}$ with elements $p_{i,t'}^{\mathring{Th}_k}$
            return ($\ell \leq p_{i,t'}^{\mathring{Th}_k}$ and $u \geq p_{i,t'}^{\mathring{Th}_k}$)    ▷ true if $p_{i,t'}^{\mathring{Th}_k} \in [\ell, u]$

        case $\neg B_{i,t'}^{\ell,u}(\varphi)$:
            $P_{t'} \leftarrow bs(\langle\mathcal{D},\mathcal{H}\rangle, t') \cdot \vec{b}(\varphi)$
            return ($\ell \geq p_{i,t'}^{\mathring{Th}_k}$ or $u \leq p_{i,t'}^{\mathring{Th}_k}$)    ▷ true if $p_{i,t'}^{\mathring{Th}_k} \notin [\ell, u]$

        case $\mathscr{B}' \wedge \mathscr{B}''$:
            return (Check($bs(\langle\mathcal{D},\mathcal{H}\rangle), \mathring{Th}_k, \mathscr{B}'$) and
                    Check($bs(\langle\mathcal{D},\mathcal{H}\rangle), \mathring{Th}_k, \mathscr{B}''$))

        case $\mathscr{B}' \vee \mathscr{B}''$:
            return (Check($bs(\langle\mathcal{D},\mathcal{H}\rangle), \mathring{Th}_k, \mathscr{B}'$) or
                    Check($bs(\langle\mathcal{D},\mathcal{H}\rangle), \mathring{Th}_k, \mathscr{B}''$))
```

4.1. A Model Checking Algorithm

A first approach of developing an algorithm to check whether a given set of belief formulae $\mathfrak{B}$ is satisfied by a given pointed doxastic system $\langle\mathcal{D},\mathcal{H}\rangle$ (i.e., checking whether $\langle\mathcal{D},\mathcal{H}\rangle$ is a model for $\mathfrak{B}$) can be obtained through a direct application of the semantics of the belief operator given in Definition 3.11. Algorithm 1 shows the resulting model checking procedure. It starts with computing the belief states for all possible evolutions of the world from $t = 1$ to t_{max}. Afterwards, it iterates through all belief formulae $\mathscr{B} \in \mathfrak{B}$ and potential pov threads $\mathring{Th}_k$ to determine whether the interpretation in the respective pov thread is able to satisfy the current belief formula. If a thread is unable to satisfy some belief formula, it is excluded from the set of potential pov threads for subsequent checks. If at least one potential pov thread remains after all belief formulae have been checked (i.e., there is at least one thread $\mathring{Th}_k$ so that all belief formulae $\mathscr{B} \in \mathfrak{B}$ are satisfied), $\langle\mathcal{D},\mathcal{H}\rangle$ is a model for $\mathfrak{B}$.

Theorem 4.1 (Soundness and completeness of Algorithm 1)**.** *The decision procedure Algorithm 1 is sound and complete and therefore a model checking procedure for PDT Logic.*

Proof. Since the presented algorithm is essentially an inductive application of Definition 3.16, it is easy to see that it yields a sound and complete decision procedure for PDT Logic. Basic belief formulae ($B_{i,t'}^{\ell,u}(F_t)$ and $B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F,G))$) return satisfiability results by directly using the respective semantic definitions from (3.10) and (3.11) as calculation rules. For every possible compound belief formula of PDT Logic ($\neg B_{i,t'}^{\ell,u}(\varphi)$, $B_{i,t'}^{\ell,u}(\mathscr{B})$, $\mathscr{B}' \wedge \mathscr{B}''$, and $\mathscr{B}' \vee \mathscr{B}''$), the procedure provides an appropriate rule according to Definition 3.16 to break down these formulae iteratively until base formulae are obtained, which can be decided as above. $\square$

The asymptotic complexity of Algorithm 1 depends on the number of belief operators $B_{i,t'}^{\ell,u}(\varphi)$ contained in $\mathfrak{B}$:

Theorem 4.2 (Time complexity of Algorithm 1)**.** *Let $\mathfrak{B}$ be a set of belief formulae and let k be the number of belief operators contained within $\mathfrak{B}$. Then, using Algorithm 1 to check whether a given pointed doxastic system $\langle \mathcal{D}, \mathcal{H} \rangle$ with m threads is a model for $\mathfrak{B}$ has time complexity $\mathcal{O}(k \cdot m)$.*

Proof. For a given pds with m threads and k belief formulae in $\mathfrak{B}$, the main procedure calls the check function at most $m \cdot k$ times. If $\mathscr{B}$ is a base formula with only a single belief operator $B_{i,t'}^{\ell,u}(\varphi)$, a single call of the check function will return a result. Otherwise, if a belief formula $\mathscr{B}$ contains more than one belief operator $B_{i,t'}^{\ell,u}(\varphi)$, the check function will be called recursively, until base formulae are obtained. Thus, for k belief operators in $\mathfrak{B}$, the satisfaction checks are performed at most $k \cdot m$ times, yielding a time complexity of $\mathcal{O}(k \cdot m)$. $\square$

From Theorem 4.2 we immediately obtain a complexity result for the model checking problem in PDT Logic:

Corollary 4.3 (Complexity of model checking for PDT Logic)**.** *The model checking problem for PDT Logic is in PTIME.*

This result shows that model checking of a set of belief formulae w.r.t. a given pointed doxastic system can be done in polynomial time. If a fully specified pds (and thereby an exhaustive specification of the set of possible threads $\hat{\mathcal{T}}$) is given, this result shows that Algorithm 1 presents a tractable procedure to perform the model checking task. However, this approach has a significant drawback as it assumes an exhaustive specification of $\hat{\mathcal{T}}$ together with precise prior probability assignments $\mathcal{I}(\hat{\mathcal{T}})$. Although there are some problem domains that actually come with such a specification (e.g., cf. the cyber security scenario described in the introduction, which we will formally treat in Chapter 5), this assumption renders Algorithm 1 infeasible for most problem domains. To overcome this problem, we will proceed with discussing a different approach, which enables satisfiability checking without requiring a specification of exact probabilities. Moreover, we show how representative threads with respect to a set of belief formulae $\mathfrak{B}$ can be constructed automatically, so that positive satisfiability results can potentially be obtained without requiring a full materialization of all possible threads $\hat{\mathcal{T}}$.

4.2. A Compact Problem Specification

Up until now we used a (pointed) doxastic system to specify a problem domain for model checking a set of belief formulae $\mathfrak{B}$ in PDT Logic. In the following sections, we show how we can reformulate the problem such that an extended set of belief formulae together with a value for t_{max} is used. The main idea of this approach is that background knowledge regarding the target domain is not given through an explicit specification of possible threads and according probabilities, but instead through sets of rules in $\mathfrak{B}$ that describe how the target domain may evolve over time. This approach has several advantages: In most scenarios, compared to requiring an exhaustive set of possible threads, specifying a set of rules (which can be expressed as prior beliefs) gives a more natural means of specifying background knowledge of the problem domain (e.g., cf. Example 3.2, which actually starts with a verbal description of rules and only later introduces the corresponding set of possible threads). Furthermore, using a set of rules to describe a problem domain is a fairly established approach and therefore this approach will provide options to simplify transformation of existing problem specifications into PDT Logic. Finally, since the set of possible threads grows exponentially with every additional time point in the set of time points τ and every additional ground atom of the language $\mathcal{L}$, an exhaustive problem specification through the set of possible threads quickly becomes infeasible, while the same situation could be described succinctly through a small set of rules. Even though such a succinct specification shifts the exponential nature of this problem from the required input specification to computational efforts, we show that the exponential effect can be curtailed with heuristics when constructing possible threads automatically.

4.2.1. Identification of Key Parameters from a Set of Belief Formulae

To simplify the following discussion, we will restrict temporal rules to only use the point frequency function *pfr*. Recall that point frequency functions are used to specify that some event F is followed by another event G after *exactly* Δt time points, while existential frequency functions *efr* are used to specify that some event F is followed by another event G *within* a time interval Δt. If existential frequency functions are required to specify a problem domain, we can rewrite them as disjunctions of point frequency functions, as the following proposition shows. If further frequency functions are defined, the presented techniques can be easily adapted.

Proposition 4.4 (*efr* rewriting)**.** *An existential frequency function efr can be equivalently represented as a disjunction of point frequency functions pfr:*

$$r^{efr}_{\Delta t}(F,G) \equiv \bigvee_{\tilde{\Delta}t:\ 0\leq\tilde{\Delta}t\leq\Delta t} r^{pfr}_{\tilde{\Delta}t}(F,G)$$

Recall that, according to Definitions 3.12 and 3.14, the specification of a pds consists of a set of agents $\mathcal{A}$, a set of threads $\hat{\mathcal{T}}$, a set of frequency functions $\mathcal{F}$, a matrix of prior probability distributions $A_0^{|\mathcal{A}|\times|\hat{\mathcal{T}}|}$, and a set of time-stamped observations $\mathcal{H}$.

Since we will only use point frequency functions in the following, the set of frequency functions $\mathcal{F}$ is always fixed to $\{pfr\}$, and thus there is no need to specify this set separately.

Instead of explicitly specifying the set of agents $\mathcal{A}$, we can just determine it from the belief expressions $B^{\ell,u}_{i,t'}(\varphi)$ contained in the set of belief formulae $\mathfrak{B}$. With a slight abuse of notation, we use $B^{\ell,u}_{i,t'}(\varphi) \in \mathfrak{B}$ to denote that belief operator $B^{\ell,u}_{i,t'}(\varphi)$ appears somewhere in a set of belief formulae $\mathfrak{B}$. Then, we can define the set of agents $\mathcal{A}_{\mathfrak{B}}$ specified through a set of belief formulae $\mathfrak{B}$ as

$$\mathcal{A}_{\mathfrak{B}} = \{i\ : B^{\ell,u}_{i,t'}(\varphi) \in \mathfrak{B}\} \tag{4.2}$$

Generally, it is possible that the explicit specification of the set of agents $\mathcal{A}$ is larger than the set $\mathcal{A}_{\mathfrak{B}}$. However, it is obvious that if no beliefs are expressed for some agent i (i.e., $i \in \mathcal{A}$ and $i \notin \mathcal{A}_{\mathfrak{B}}$), this agent will not influence satisfiability checking results whatsoever. Thus, this agent can simply be disregarded and, consequently, it suffices to use the set $\mathcal{A}_{\mathfrak{B}}$.

Similarly, instead of specifying the set of ground atoms of the language $\mathcal{L}$ through the sets of predicates $\mathcal{L}_{pred}$ and constants $\mathcal{L}_{cons}$, we can define a set of event formulae $\mathbf{F}_{\mathfrak{B}}$ representing all belief objects occurring in a set of belief formulae $\mathfrak{B}$ as

$$\mathbf{F}_{\mathfrak{B}} = \Big\{F : \big(B^{\ell,u}_{i,t'}(F_t) \in \mathfrak{B}\ \vee\ B^{\ell,u}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(F,G)) \in \mathfrak{B}\ \vee\ B^{\ell,u}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(G,F)) \in \mathfrak{B}\big)\Big\}. \tag{4.3}$$

This definition gives rise to a potential definition of the set of possible worlds Ω as the Herbrand base $\mathcal{B}^{\mathbf{F}_{\mathfrak{B}}}$ of $\mathbf{F}_{\mathfrak{B}}$ (resp. the set of admissible worlds $\hat{\Omega} \subseteq \Omega$ complying with Definition 3.5). However, as we will show later, there are more options to constrain the sets of possible worlds to allow for a more concise problem representation.

Note that according to Definition 3.2, formulae may include both atoms and observation atoms. Consequently, $\mathbf{F}_{\mathfrak{B}}$ does not only specify ontic facts of possible worlds, but also possible observations of these ontic facts. With this approach, occurrences of observations are limited to the ones specified in $\mathbf{F}_{\mathfrak{B}}$. This can be seen as the specification of a sensor model for groups of agents $\mathcal{G} \subseteq \mathcal{A}_{\mathfrak{B}}$.

Remark 4.1. A strict application of (4.3) would prohibit simple specifications of group observations $Obs_{\mathcal{G}}(l)$ with $|\mathcal{G}| > 1$ in $\mathfrak{B}$. To ensure that the set of admissible worlds actually contains worlds with $Obs_{\mathcal{G}}(l)$, a full specification of such an observation as $\bigwedge_{\mathcal{G}' \subseteq \mathcal{G}} Obs_{\mathcal{G}'}(l)$ in $\mathfrak{B}$ would be required (otherwise there might be no world $\omega \in \mathcal{B}^{\mathbf{F}_{\mathfrak{B}}}$ with $\omega \models Obs_{\mathcal{G}}(l)$ that satisfies the second property in the definition of possible worlds (cf. Definition 3.5)). However, the required full specification of an observation for admissible worlds can be determined solely through the simple observation specification $Obs_{\mathcal{G}}(l)$. In order to keep the specification of $\mathfrak{B}$ as compact as possible, we allow for simple specifications $Obs_{\mathcal{G}}(l)$ and assume that they are expanded with $\bigwedge_{\mathcal{G}' \subset \mathcal{G}} Obs_{\mathcal{G}'}(l)$ while creating $\mathbf{F}_{\mathfrak{B}}$.

An alternative approach would be to construct $\mathbf{F}_{\mathfrak{B}}$ only through ontic facts appearing in $\mathfrak{B}$ and create a set of admissible worlds by combining all ontic facts with all possible admissible observations w.r.t. Definition 3.5. These approaches differ in the requirements of observation specifications: the former requires to specify every possible observation explicitly, while the latter requires to exclude every impossible observation explicitly. Since in most scenarios the set of observations actually being possible (w.r.t. the problem domain) is significantly smaller than the set of all admissible observations, the presented approach will usually yield a more compact problem specification. If desired, one could employ the latter approach instead without impacting the functionality of the following methods.

Background knowledge regarding the target domain—that was given through an explicit representation of possible threads before—can now also be specified as prior beliefs (i.e., beliefs $B_{i,0}^{\ell,u}(\varphi)$) in $\mathfrak{B}$. Recall from Section 3.2.4 that we assume a commonly known prior distribution $\mathring{\mathcal{I}}_{i,t}^{Th}$ which is equal for all agents $i \in \mathcal{A}_{\mathfrak{B}}$. As the belief semantics is defined with respect to the probabilistic interpretations $\mathring{\mathcal{I}}_{i,t'}^{Th}$ (cf. Definition 3.11), it follows that every prior belief $B_{i,0}^{\ell,u}(\varphi)$ is common knowledge as well. Consequently, we can express background knowledge as prior beliefs of any arbitrary agent $i \in \mathcal{A}_{\mathfrak{B}}$.

As pointed out in Chapter 3, satisfiability of beliefs in temporal rules $B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F,G))$

with certain properties are independent of the respective set of threads $\hat{\mathcal{T}}$ or the associated interpretation $\mathcal{I}(\hat{\mathcal{T}})$ (cf. Remark 3.5): if the respective frequency function corresponds to FF1, FF2, or FF3 of Definition 3.10 (i.e., F is a contradiction, G is a tautology, or F is a tautology and G is a contradiction), beliefs are either trivially satisfied for quantifications with $u = 1$ (resp. $\ell = 0$) or generally unsatisfiable. In the former case, trivially satisfiable beliefs can be disregarded without influencing satisfiability results, while for the latter case satisfiability checking can terminate immediately with a negative result. Thus, in the following we assume that $\mathfrak{B}$ contains only beliefs in rules that do not correspond to frequency function axioms FF1-FF3.

Example 4.1 (Trains revisited)**.** An informal verbal description of the train problem was given in Example 3.2 with a corresponding formal specification through a set of possible threads $\hat{\mathcal{T}}$ in Example 3.4 and probability assignments in Example 3.6. Using the above considerations on the expression of background knowledge as beliefs in rules, we can reformulate the verbal rules given in Example 3.2 together with the probabilistic information from Example 3.6 as a set of formal beliefs $\mathfrak{B}$ with according explanations below:

$$\mathfrak{B} = \left\{ \begin{array}{lll}
\mathscr{B}_1 = & B^{1,1}_{A,0}\big(at(T_1, C_A)_1\big) \wedge B^{1,1}_{A,0}\big(on(A, T_1)_1\big), & \\
\mathscr{B}_2 = & B^{.81,.81}_{A,0}\big(r^{pfr}_0(at(T_1, C_A), punct(T_1))\big) & (\mathscr{B}_2') \\
 & \wedge\ B^{.81,.81}_{A,0}\big(r^{pfr}_0(at(T_2, C_C), punct(T_2))\big), & (\mathscr{B}_2'') \\
\mathscr{B}_3 = & B^{1,1}_{A,0}\big(r^{pfr}_3(\ punct(T_1) \wedge at(T_1, C_A),\ at(T_2, C_C) \wedge on(A, T_2))\big) & (\mathscr{B}_3') \\
 & \wedge\ B^{1,1}_{A,0}\big(r^{pfr}_5(\neg punct(T_1) \wedge at(T_1, C_A),\ at(T_2, C_C) \wedge on(A, T_2))\big), & (\mathscr{B}_3'') \\
\mathscr{B}_4 = & B^{1,1}_{A,0}\big(r^{pfr}_2(\ punct(T_2) \wedge at(T_2, C_C),\ at(T_2, C_B) \wedge on(A, T_2))\big) & (\mathscr{B}_4') \\
 & \wedge\ B^{1,1}_{A,0}\big(r^{pfr}_3(\neg punct(T_2) \wedge at(T_2, C_C),\ at(T_2, C_B) \wedge on(A, T_2))\big), & (\mathscr{B}_4'') \\
\mathscr{B}_5 = & B^{1,1}_{A,0}\big(r^{pfr}_0(\neg punct(train) \wedge at(train, city), Obs_{\{A\}}(\neg punct(train)))\big), & \\
\mathscr{B}_6 = & B^{.93,.93}_{A,0}\big(r^{pfr}_2(Obs_{\{A\}}(\neg punct(train)), Obs_{\{AB\}}(\neg punct(train)))\big), & \\
train \in \{T_1, T_2\}, & & \\
city \in \{C_A, C_B\} & &
\end{array} \right\}$$

Note that all beliefs are expressed for time $t = 0$, i.e., these are prior beliefs that are by definition commonly known among all agents. All beliefs expressed in this example are assigned to A, but they could equivalently be assigned to B or to both.

$\mathscr{B}_1$ states that train T_1 is at city C_A at time $t = 1$ and that Alice is on that train. $\mathscr{B}_2$ states that both agents believe that trains are punctual (denoted by $punct(train)$) with a probability of 0.81. The probability values in this example are obtained by summing over the probabilities given in Example 3.6 for all threads given in Example 3.4 where the respective belief object is satisfied. To have an equivalent representation of the previous example, we use exact probability values (i.e., $\ell = u$) instead of intervals. Note that $punct(train)$ is an additional predicate with a variable $train$ that helps to formulate the background knowledge in a concise way. Formula $\mathscr{B}_2$ does not yet specify what the consequences of a non-punctual train are, only that a train is expected to be punctual with a certain probability. $\mathscr{B}_3$ states that Alice is able to board train T_2 after three time steps if train T_1 is punctual and that Alice has to wait for two additional time points otherwise. $\mathscr{B}_4$ states that train T_2 will arrive at city C_B two time points after being in city C_C. Otherwise she will arrive one time point later. $\mathscr{B}_5$ states that Alice will always notice when her train leaves a city not punctually. This is an example for a sensor model specification as discussed above. Finally, $\mathscr{B}_6$ states that Alice will call Bob with a probability of 0.93 if her train is not punctual.

Example 4.2 (Trains continued). With the definition of the set of belief formulae $\mathfrak{B}$ from the above example, we can now also specify the set of event formulae $\mathbf{F}_{\mathfrak{B}}$ required to model the possible scenarios described through $\mathfrak{B}$:

$$\mathbf{F}_{\mathfrak{B}} = \left\{ \begin{array}{l} at(T_1, C_A),\ at(T_1, C_B),\ at(T_2, C_B),\ at(T_2, C_C), \\ on(A, T_1),\ on(A, T_2),\ punct(T_1),\ punct(T_2), \\ Obs_{\{A\}}(\neg punct(T_1)),\ Obs_{\{AB\}}(\neg punct(T_1)), \\ Obs_{\{A\}}(\neg punct(T_2)),\ Obs_{\{AB\}}(\neg punct(T_2)) \end{array} \right\}$$

To simplify the following discussion, we assume that conjunctive formulae $\mathscr{B} = \mathscr{B}' \wedge \mathscr{B}'' \in \mathfrak{B}$ are replaced with individual formulae of the respective conjuncts: $\mathfrak{B} = \mathfrak{B} \setminus \{\mathscr{B}\} \cup \{\mathscr{B}', \mathscr{B}''\}$. This does not impact the satisfiability checking properties of $\mathfrak{B}$ because all formulae in $\mathfrak{B}$ have to be satisfied simultaneously in order to return a positive result and thus, both $\mathscr{B}'$ and $\mathscr{B}''$ have to be satisfied, regardless of their representation as two individual formulae or as one conjunction.

Now, what remains to be determined is the set of threads $\hat{\mathcal{T}}$, a corresponding prior probability distribution $\mathcal{I}(\hat{\mathcal{T}})$ (resp. a matrix of prior probability distributions $A_0^{|\mathcal{A}|\times|\hat{\mathcal{T}}|}$, where every row is formed by $\mathcal{I}(\hat{\mathcal{T}})$), and possibly a set of time-stamped observation atoms $\mathcal{H}$. The tasks of determining $\hat{\mathcal{T}}$ and $\mathcal{H}$ can be treated jointly: since the set of relevant threads needs to be determined anyway, we simply create $\hat{\mathcal{T}}$ such that $\hat{\mathcal{T}} \models \mathcal{H}$.

In the next section we will show how we can transform a set of PDT Logic belief formulae $\mathfrak{B}$ together with a given set of threads $\hat{\mathcal{T}}$ into a linear program in order to determine satisfiability of $\mathfrak{B}$ with respect to $\hat{\mathcal{T}}$. Afterwards, we will discuss how a suitable set of threads $\hat{\mathcal{T}}$ to represent the information contained in $\mathfrak{B}$ can be constructed automatically. Using these results, it is possible to model a problem domain in PDT Logic solely through a set of belief formulae $\mathfrak{B}$ together with the specification of a maximum time point t_{max}. All other key parameters of the domain—such as the set of agents and the set of ground atoms—can be extracted from $\mathfrak{B}$ automatically.

4.3. Representing the Satisfiability Problem as a Linear Program

The considerations from the previous section show that most parameters for a problem specification can be extracted from a given set of belief formulae $\mathfrak{B}$. In this section, we assume that only a set of belief formulae $\mathfrak{B}$ together with a set of possible threads $\hat{\mathcal{T}}$ is given. $\mathfrak{B}$ is then satisfiable with respect to $\hat{\mathcal{T}}$ (denoted by $sat(\mathfrak{B}, \hat{\mathcal{T}})$) if a prior interpretation $\mathcal{I}(\hat{\mathcal{T}})$ can be found such that all belief formulae in $\mathfrak{B}$ are satisfied. By extracting linear constraints on $\mathcal{I}(\hat{\mathcal{T}})$ from $\mathfrak{B}$, we show how the satisfiability problem can be transformed into a linear program. Checking satisfiability of $\mathfrak{B}$ with respect to $\hat{\mathcal{T}}$ is then equivalent to checking whether the corresponding linear program has a feasible solution.

For a given set of threads with an unknown prior interpretation $\mathcal{I}(\hat{\mathcal{T}})$, the satisfiability checking task significantly increases in complexity compared to the model checking task. Formulation of the satisfiability checking problem in Definition 4.1 might be somewhat delusive: As the existence of a single thread in the context of some interpretation suffices to verify satisfiability of a set of belief formulae $\mathfrak{B}$, it appears intuitive to develop a method to construct such a thread—if possible—and neglect the other threads, or, vice versa, start with the entire set of threads $\hat{\mathcal{T}}$ and iteratively prune all threads that fail to satisfy any formula from $\mathfrak{B}$. In fact, such a pruning approach was used in Algorithm 1 to check whether a given set of threads is a model for a set of belief formulae. Unfortunately, these approaches are inapplicable if the prior interpretation is unknown. As the semantics of belief operators (cf. Definition 3.11) relies on subjective posterior probabilistic interpretations (i.e., on probability assignments for *multiple* threads), it is generally not possible to find a single thread $\mathring{Th}$ satisfying the satisfiability checking problem from Definition 4.1 without determining probabilities for all threads. Vice versa, it is generally not possible to discard any thread, because determining whether it satisfies any belief formula can only be done if its respective probability assignment is known. Instead, we will show that belief formulae can equivalently be expressed as sets of linear constraints on the unknown prior interpretation $\mathcal{I}(\hat{\mathcal{T}})$. Then, checking satisfiability of $\mathfrak{B}$ is equivalent to checking whether there is a possible assignment to $\mathcal{I}(\hat{\mathcal{T}})$ so that all constraints are satisfied.

We will use x_k to denote the unknown prior probability of thread Th_k, i.e., if $\hat{\mathcal{T}}$ contains m threads, then its unknown prior probability assignment is represented as

$$\mathcal{I}(\hat{\mathcal{T}}) = \left(x_1, \quad \cdots, \quad x_m\right)^T. \tag{4.4}$$

The goal of the following methods is to provide constraints on the x_k so that all belief formulae $\mathscr{B} \in \mathfrak{B}$ are satisfied. Since these variables represent a probability distribution over the set of threads, there are two obvious constraints to begin with:

$$0 \leq x_k \leq 1, \ \forall k \in \{1, ..., m\} \tag{4.5}$$

and

$$\sum_{k=1}^{m} x_k = 1 \tag{4.6}$$

4.3.1. Representation of Subjective Posterior Probabilities

Since the semantics of beliefs is defined in terms of the respective agents' subjective probability assignments in the respective pov thread, we need means to express the subjective posterior probabilistic interpretations $\mathcal{I}_{i,t'}^{\mathring{Th}}$ of an agent i in terms of the prior probability values x_k. These interpretations change at a time point t whenever an observation $Obs_{\{i\}}(l)_t$ is possible for agent i. If an observation is possible for an agent, we can partition the set of threads into two sets: one partition containing the set of threads where agent i does observe the respective fact l and one partition where agent i does not observe the respective fact. The subjective probability assignments need to be updated within each partition to reflect this information about observation occurrences: Taking every thread within a partition as a possible pov thread, the probability assignments for all other threads within this partition need to be scaled according to the update rule in Definition 3.9 and the pov thread specific probability assignments for all threads outside of the respective partition need to be set to zero.

Generally, this leads to one vector of subjective probabilities over all threads for every possible pov thread (cf. the Definition of belief states in (3.17)). However, we can leverage the semantic properties of PDT Logic to obtain a parsimonious representation of the updated subjective probabilities without representing every pov thread explicitly. Note that all threads within one partition as described above are indistinguishable to agent i at the respective time point (i.e., all threads within one partition exhibit exactly the same set of observations for agent i up to time point t) and therefore receive the same probability assignment for every possible pov thread within this partition (cf. Proposition 3.6). Consequently, the updated probability assignments for every thread in $\hat{\mathcal{T}}$ can receive only one of two different types of value assignments: a scaled version of the thread's previous probability assignment according to Definition 3.9, or zero, depending on whether the agent actually observes the fact l or not. The following proposition shows that we do not need to consider the cases with zero probabilities in order to perform satisfiability checking tasks.

Proposition 4.5 (Irrelevance of zero-interpretations)**.** *Let $\mathcal{I}_{i,t'}^{\mathring{Th}}$ be the subjective posterior probability interpretation at time t' for some agent i in pov thread $\mathring{Th}$ (i.e., this interpretation is determined through the prior interpretation and interpretation updates corresponding to pov thread $\mathring{Th}$). If this interpretation assigns a probability of zero to some thread Th (i.e., $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 0$), then satisfiability of any subsequent nontrivial belief $B_{i,t''}(\varphi)$ with $t'' > t'$ is independent of $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th)$.*

Proof. Every belief $B_{i,t'}^{\ell,u}(\varphi)$ with $\ell > 0$ in a fact or in another belief (i.e., $\varphi = F_t$ or $\varphi = B_{j,t}^{\ell_j,u_j}(\cdot)$) requires that there needs to be at least one thread Th with a nonzero probability such that $Th \models \varphi$. Therefore, a thread Th with $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 0$ can clearly not prove satisfiability of a belief $B_{i,t''}^{\ell,u}(\varphi)$ with $t'' \geq t'$. A negative satisfiability result (i.e., $\mathfrak{B}$ is unsatisfiable w.r.t. $\hat{\mathcal{T}}$) cannot be obtained from such a zero assignment either, because any consistent interpretation (i.e., the probability assignments of all threads sum to one) needs to assign a nonzero probability to at least one thread, which could then possibly satisfy the belief. The same considerations hold for beliefs $B_{i,t'}^{\ell,u}(\varphi)$ with $\ell = 0$ and $u < 1$: Although a thread with $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 0$ satisfies the lower bound $\ell = 0$, the upper bound $u < 1$ requires the existence of another thread Th' with a nonzero probability $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th') > 0$ such that $Th' \models \neg\varphi$. Consequently, $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 0$ can only prove satisfiability of beliefs $B_{i,t'}^{\ell,u}(\varphi)$ with $\ell = 0$ and $u = 1$. These are trivial beliefs that are satisfied by every thread and every possible probability assignment and thus, their satisfiability can be proven without $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 0$, too.

Analogous considerations hold for beliefs in rules: A belief $B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F,G))$ with $\ell > 0$ requires the existence of a thread with a nonzero probability such that $\mathsf{fr}(Th, F, G, \Delta t) > 0$, and thus a thread Th with $\mathcal{I}_{i,t'}^{\mathring{Th}}(Th) = 0$ cannot prove satisfiability of this belief. Satisfiability of a belief $B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F,G))$ with $\ell = 0$ and $u < 1$ depends on the respective frequencies $\mathsf{fr}(Th', F, G, \Delta t)$ in additional threads Th' with nonzero probabilities. □

As a result of this proposition, we can merge the nonzero entries from both cases (agent i observes the fact l and agent i does not observe the fact l) into a single probability distribution vector for each agent i and time point t. This yields a modified version of the update rule from Definition 3.9. We will use this modified update rule to determine linear constraints on the unknown prior probabilities x_k.

Definition 4.2 (Modified update rule)**.** Let i be an agent, t' be a time point where some observation $Obs_{\{i\}}(l)$ can occur and Th be a thread. Then, a compressed subjective posterior probability assignment $\check{\mathcal{I}}_{i,t'}(Th)$ for agent i at time t' for thread Th is given through

$$\check{\mathcal{I}}_{i,t'}(Th) = \frac{1}{\alpha_{i,t'}^{Th}} \cdot \check{\mathcal{I}}_{i,t'-1}(Th) \tag{4.7}$$

with $\alpha_{i,t'}^{Th}$ again being a normalization factor to ensure that the probabilities of all threads that agent i considers possible sum to one:

$$\alpha_{i,t'}^{Th} = \sum_{Th'(t') \in \mathcal{K}_i(Th(t'))} \check{\mathcal{I}}_{i,t'}(Th')$$

Example 4.3 (Modified update rule)**.** To illustrate the modified update rule, we return to the situation described in Example 3.7. In this example we assumed that train T_1 is running late and A does not inform B about it. This resulted in the following updated interpretation for A:

$$\mathring{\mathcal{I}}_{A,3}^{Th_4} = \mathring{\mathcal{I}}_{A,3}^{Th_6} = \mathring{\mathcal{I}}_{A,3}^{Th_8} = \begin{pmatrix} 0 & 0 & 0 & 0.4 & 0 & 0.2 & 0 & 0.4 & 0 \end{pmatrix}$$

In the given example, two additional hypothetical partitions of the set of threads $\hat{\mathcal{T}}$ are possible for Alice at time point $t = 3$. If train T_1 is running late and A does inform B about it, threads Th_5, Th_7, and Th_9 are indistinguishable to A, yielding the updated subjective interpretation

$$\mathring{\mathcal{I}}_{A,3}^{Th_5} = \mathring{\mathcal{I}}_{A,3}^{Th_7} = \mathring{\mathcal{I}}_{A,3}^{Th_9} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0.14 & 0 & 0.65 & 0 & 0.21 \end{pmatrix}$$

If T_1 is on time, Alice considers threads Th_1, Th_2, and Th_3 as possible. The corresponding subjective interpretation is then

$$\mathring{\mathcal{I}}_{A,3}^{Th_1} = \mathring{\mathcal{I}}_{A,3}^{Th_2} = \mathring{\mathcal{I}}_{A,3}^{Th_3} = \begin{pmatrix} 0.86 & 0.03 & 0.11 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

These three different subjective interpretations have nonzero entries exactly for the threads that are in the partitions of the respective pov thread. Since the partitions are not overlapping, we can merge the nonzero entries into a single probability vector

$$\check{\mathcal{I}}_{A,3} = \begin{pmatrix} 0.86 & 0.03 & 0.11 & 0.4 & 0.14 & 0.2 & 0.65 & 0.4 & 0.21 \end{pmatrix}.$$

Note that in this modified update rule, the update for each pov thread does not specify interpretations over all threads anymore, but instead only the reflexive interpretations for each thread Th, given that Th is the pov thread, are used. As discussed above, for the satisfiability problem this is still a sufficient representation of posterior probabilities, because all other potential pov threads $\mathring{Th}$ in the respective partition are indistinguishable to agent i and therefore yield exactly the same interpretations. It should be noted however that $\check{\mathcal{I}}_{i,t'}(Th)$ is *not* a probabilistic vector anymore, i.e., its elements do not sum to one. Compared to the representation of belief states from Section 3.3.1, information about distinguishable worlds is lost. Thus, reconstruction of an agent's belief state from this representation is only possible with an additional specification of the respective relations $\mathcal{K}_i$.

Returning to the problem representation from (4.4), we can use the modified update rule to obtain an inductive definition of subjective posterior probabilities based on the respective (unknown) prior probabilities x_k. If $\mathcal{I}(\hat{\mathcal{T}}) = \begin{pmatrix} x_1, & \cdots, & x_m \end{pmatrix}^T$ is the prior interpretation over the set of threads, agent i's compressed subjective posterior interpretations $\check{\mathcal{I}}_{i,t'}$ at the time point t' of the first possible observation can be represented as

$$\check{\mathcal{I}}_{i,t'}(\hat{\mathcal{T}}) = \begin{pmatrix} \frac{1}{\alpha^1_{i,t'}} \cdot x_1, & \cdots, & \frac{1}{\alpha^m_{i,t'}} \cdot x_m \end{pmatrix}^T, \tag{4.8}$$

with the update factors $\alpha^k_{i,t'}$ determined through

$$\begin{pmatrix} \alpha^1_{i,t'} & \cdots & \alpha^m_{i,t'} \end{pmatrix}^T = \begin{pmatrix} x_1, & \cdots, & \cdot x_m \end{pmatrix} \cdot \begin{pmatrix} \kappa^{i,t'}_{1,1} & \cdots & \kappa^{i,t'}_{1,m} \\ \vdots & \ddots & \vdots \\ \kappa^{i,t'}_{m,1} & \cdots & \kappa^{i,t'}_{m,m} \end{pmatrix},$$

$$\kappa^{i,t'}_{j,k} = \begin{cases} 1 & \text{if } Th_k(t') \in \mathcal{K}_i(Th_j(t')) \\ 0 & \text{if } Th_k(t') \notin \mathcal{K}_i(Th_j(t')) \end{cases}$$

with a (symmetric) matrix of indicators $\kappa^{i,t'}_{j,k}$ denoting whether agent i considers thread Th_k possible in thread Th_j at time t'. Using (4.8) as the base case, we can then define interpretation updates for the next possible observation at time t'' inductively as

$$\check{\mathcal{I}}_{i,t''}(\hat{\mathcal{T}}) = \begin{pmatrix} \frac{1}{\alpha^1_{i,t''}} \cdot \frac{1}{\alpha^1_{i,t'}} \cdot x_1, & \cdots, & \frac{1}{\alpha^m_{i,t''}} \cdot \frac{1}{\alpha^m_{i,t'}} \cdot x_m \end{pmatrix}^T \tag{4.9}$$

To simplify notation, in the following we use a single factor $a^k_{i,t'}$ to represent the aggregated sequence of scaling factors $(\alpha^k_{i,t_1} \cdot \alpha^k_{i,t_2} \cdot ...)$ for all observations that can occur at time points $t_1, t_2, ...$ between $t = 1$ and $t = t'$ for agent i, i.e., agent i's subjective posterior interpretations $\check{\mathcal{I}}_{i,t'}(\hat{\mathcal{T}})$ at time t' are given as

$$\check{\mathcal{I}}_{i,t'}(\hat{\mathcal{T}}) = \left(a^1_{i,t'} \cdot x_1, \quad \cdots, \quad a^m_{i,t'} \cdot x_m\right)^T . \tag{4.10}$$

Note that potential interpretation updates for an agent i can occur at some time point t if and only if some observation $Obs_{\{i\}}(l)$ is possible at that time point. Hence, for any time interval between two possible observations, the subjective interpretations are constant:

Proposition 4.6 (Piecewise constant interpretations). *Let t_1 and t_2 with $t_1 < t_2$ be two time points such that observations for an agent i are possible at t_1 and t_2, but at no time point t in between t_1 and t_2. Then, the compressed subjective interpretation $\check{\mathcal{I}}_{i,t'}(\hat{\mathcal{T}})$ is constant for all time points $t_1 \leq t < t_2$:*

$$\forall t \in [t_1, t_2 - 1] : \ \check{\mathcal{I}}_{i,t}(\hat{\mathcal{T}}) = \check{\mathcal{I}}_{i,t_1}(\hat{\mathcal{T}})$$

This proposition states that all constraints identified in the following section do not only restrict the subjective interpretations at single time points, but instead restrict the interpretations for the respective time interval between any two possible observations.

4.3.2. Extracting Linear Constraints from Belief Formulae

Now that we have established representation (4.10) of subjective posterior interpretations in terms of the unknown prior probabilities x_k, we can use this representation to extract linear constraints on the x_k from the set of belief formulae $\mathfrak{B}$.

We assume that the distributive property of the belief operator from Lemma 3.7 is applied whenever possible, i.e., belief formulae $B^{\ell,u}_{i,t'}(\mathscr{B}_1 \otimes \mathscr{B}_2)$ with $\otimes \in \{\wedge, \vee\}$ are separated into $B^{\ell,u}_{i,t'}(\mathscr{B}_1) \otimes B^{\ell,u}_{i,t'}(\mathscr{B}_2)$. Furthermore, without loss of generality, we can assume that conjunctive formulae $\mathscr{B} = \mathscr{B}_1 \wedge \mathscr{B}_2$ are replaced through $\mathfrak{B} \setminus \{\mathscr{B}\} \cup \{\mathscr{B}_1, \mathscr{B}_2\}$ and that trivial beliefs (with $\ell = 0$ and $u = 1$) are removed from $\mathfrak{B}$.

Moreover, we assume that all belief formulae $\mathscr{B} \in \mathfrak{B}$ are represented in negation normal form (NNF), i.e., the negation operator is only applied to atoms. Since any arbitrary logic formula can equivalently be expressed as a formula in NNF (cf. e.g., [BEL+01]), this assumption does not restrict $\mathfrak{B}$ either.

With these assumptions, the following types of belief formulae $\mathscr{B}$ can occur in $\mathfrak{B}$:

- atomic belief formulae $\mathscr{B} = B^{\ell,u}_{i,t'}(\varphi)$
- negated atomic belief formulae $\mathscr{B} = \neg B^{\ell,u}_{i,t'}(\varphi)$

- disjunctive belief formulae $\mathscr{B} = \mathscr{B}_1 \vee \mathscr{B}_2$

For each of these types, we will now show how the respective formula can be expressed as a set of linear constraints on the prior probabilities x_k.

Atomic Belief Formulae Using the parsimonious representation of subjective posterior interpretations $\check{\mathcal{I}}_{i,t'}(Th)$ given through the modified update rule in Definition 4.2 requires an adaption when deciding satisfiability of belief formulae. Before, satisfaction of a belief formula in a given pov thread could be determined by summing over the respective subjective interpretations of all threads in which the belief object is satisfied. Threads that an agent does not consider possible anymore w.r.t. the given pov thread are automatically "excluded" as they have a probability assignment of zero. In the compressed representation, the respective probability assignments for threads considered impossible are overloaded with different probability assignments given that the agent is in another pov thread, as illustrated in Example 4.3. We obtain an adapted version of satisfiability testing by explicitly ensuring that only those interpretations of threads are summed that are still considered possible w.r.t. the respective pov thread. As this additional constraint only excludes summands with zero-values, the original semantics is still maintained. Thus, we use equivalence classes $\mathcal{C}_{i,t'} = \{\mathcal{C}^1_{i,t'}, \mathcal{C}^2_{i,t'}, ...\}$ to represent the set of distinguishable situations for agent i at time t'. Naturally, two threads Th_1, Th_2 are indistinguishable and therefore in the same equivalence class for agent i at time t', if they exhibit exactly the same observations for agent i for all time points $t \in \{1, .., t'\}$. All threads outside of a particular equivalence class receive a probability of zero for every pov thread $\mathring{Th}$ within the respective equivalence class and—as discussed in the previous section—therefore do not contribute to the satisfiability properties. Then, in the belief semantics from Definition 3.11, instead of summing over all threads $Th \in \hat{\mathcal{T}}$ with certain properties, we can restrict the range to $Th \in \mathcal{C}^k_{i,t} : (\mathring{Th} \in \mathcal{C}^k_{i,t})$ while maintaining the original semantics. Naturally, a belief formula is then satisfiable if there exists at least one equivalence class that satisfies the respective beliefs. For instance, a belief in a fact $B^{\ell,u}_{i,t'}(F_t)$ is satisfiable with respect to an agent i's compressed subjective posterior interpretation $\check{\mathcal{I}}_{i,t'}$ at time t' iff

$$\exists \mathcal{C}^k_{i,t'} \in \mathcal{C}_{i,t'} : \; \ell \leq \sum_{n:\; (Th_n \in \mathcal{C}^k_{i,t'} \wedge Th_n(t) \models F)} a^n_{i,t'} \cdot x_n \leq u \tag{4.11}$$

Such a constraint can equivalently be expressed as a set of linear inequalities with conjunctive and disjunctive connectives, leading to an alternative representation of the satisfiability problem.

Corollary 4.7 (Alternative satisfiability representation for atomic beliefs)**.** *Let* $\check{\mathcal{I}}_{i,t'}(\hat{\mathcal{T}}) = \left(a^1_{i,t'} \cdot x_1, \; \cdots, \; a^m_{i,t'} \cdot x_m\right)^T$ *and* $\check{\mathcal{I}}_{j,t}(\hat{\mathcal{T}}) = \left(a^1_{j,t} \cdot x_1, \; \cdots, \; a^m_{j,t} \cdot x_m\right)^T$ *be the com-*

pressed representation of agent i and j's respective subjective posterior probabilities at time t' and t, respectively, as given in (4.10), and let $\mathcal{C}_{i,t'}$ and $\mathcal{C}_{j,t}$ be the sets of worlds that agent i and agent j can distinguish at the respective time point. Then, an atomic belief expression $\mathscr{B}$ is satisfiable w.r.t. $\check{\mathcal{I}}_{i,t'}(\hat{\mathcal{T}})$ for

1. *belief in a fact $\mathscr{B} = B_{i,t'}^{\ell,u}(F_t)$ iff*

$$\bigvee_{\mathcal{C}^k_{i,t'} \in \mathcal{C}_{i,t'}} \left(\Big(\sum_{\substack{n:\ (Th_n \in \mathcal{C}^k_{i,t'} \\ \wedge Th_n(t) \models F)}} -a^n_{i,t'} \cdot x_n \leq -\ell \Big) \wedge \Big(\sum_{\substack{n:\ (Th_n \in \mathcal{C}^k_{i,t'} \\ \wedge Th_n(t) \models F)}} a^n_{i,t'} \cdot x_n \leq u \Big) \right) \tag{4.12}$$

2. *belief in a rule $\mathscr{B} = B_{i,t'}^{\ell,u}(r^{pfr}_{\Delta t}(F,G))$ iff*

$$\begin{aligned} \bigvee_{\mathcal{C}^k_{i,t'} \in \mathcal{C}_{i,t'}} \Bigg(& \Big(\sum_{n:\ (Th_n \in \mathcal{C}^k_{i,t'})} -a^n_{i,t'} \cdot x_n \cdot pfr(Th_n, F, G, \Delta t) \leq -\ell \Big) \\ & \wedge \Big(\sum_{n:\ (Th_n \in \mathcal{C}^k_{i,t'})} a^n_{i,t'} \cdot x_n \cdot pfr(Th_n, F, G, \Delta t) \leq \ u \Big) \Bigg) \end{aligned} \tag{4.13}$$

3. *nested belief $\mathscr{B} = B_{i,t'}^{\ell,u}(B_{j,t}^{\ell_j,u_j}(\varphi))$ iff*

$$\begin{aligned} \bigvee_{\mathcal{C}^k_{i,t'} \in \mathcal{C}_{i,t'}} \Bigg(& \Big(\sum_{\substack{n:\ (Th_n \in \mathcal{C}^k_{j,t}) \wedge \\ Th_n \models \varphi\ \wedge\ (\{\mathcal{C}^k_{j,t} \cap \mathcal{C}^k_{i,t'}\} \neq \emptyset)}} -a^n_{j,t} \cdot x_n \ \leq -\ell_j \Big) \\ & \wedge \Big(\sum_{\substack{n:\ (Th_n \in \mathcal{C}^k_{j,t}) \wedge \\ Th_n \models \varphi\ \wedge\ (\{\mathcal{C}^k_{j,t} \cap \mathcal{C}^k_{i,t'}\} \neq \emptyset)}} a^n_{j,t} \cdot x_n \ \leq \ u_j \Big) \\ & \wedge \Big(\sum_{\substack{n:\ Th_n \in \{\mathcal{C}^k_{j,t} \cap \mathcal{C}^k_{i,t'}\} \\ \wedge\ Th_n \models \varphi}} -a^n_{i,t'} \cdot x_n \ \leq -\ell \Big) \\ & \wedge \Big(\sum_{\substack{n:\ Th_n \in \{\mathcal{C}^k_{j,t'} \cap \mathcal{C}^k_{i,t'}\} \\ \wedge\ Th_n \models \varphi}} a^n_{i,t'} \cdot x_n \ \leq \ u \Big) \Bigg) \end{aligned} \tag{4.14}$$

As discussed above, the representations for satisfiability of beliefs in facts (4.12) and beliefs in rules (4.13) are obtained directly by replacing the range of threads $\mathcal{T}$ in the sum with the respective set of threads $\mathcal{C}^k_{j,t'}$ considered possible by agent i at time t'. The inequalities for nested beliefs (4.14) are obtained by ensuring in the first two lines that in every situation that agent i conceives as a possible situation for agent j (expressed through the constraint $n: \ (Th_n \in \mathcal{C}^k_{j,t}) \wedge \mathcal{C}^k_{j,t} \cap \mathcal{C}^k_{i,t'} \neq \emptyset)$, agent j's belief in the respective fact φ (expressed through the constraint $Th_n \models \varphi$) is within $[\ell_j, u_j]$. The latter two lines ensure that for these respective situations, the outer belief of agent i is satisfied, as well. Note that the belief object φ in (4.14) might contain additional belief operators, i.e., beliefs with multiple levels of nesting are expressed. In this case, evaluation of $Th \models \varphi$ in the first two lines of (4.14) yields additional constraints of type (4.12)–(4.14), such that the formula is evaluated recursively.

Negated Atomic Belief Formulae To satisfy a negated atomic belief formula $\mathscr{B} = \neg B^{\ell,u}_{i,t'}(\varphi)$, the accumulated probabilities of all threads that satisfy the belief object φ in an equivalence class $\mathcal{C}^k_{i,t'}$ must be either lower than ℓ or higher than u, i.e., the individual disjuncts in (4.12)–(4.14) have to be negated. By pushing the negations inward and using $\sum_{\cdots}(\cdots)$ as a representative for the respective sums defined in (4.12) and (4.13) to express satisfiability of atomic beliefs, we can represent negations of the according beliefs expressed in (4.12) and (4.13) as

$$\bigvee_{\mathcal{C}^k_{i,t'} \in \mathcal{C}_{i,t'}} \left(\Big(-\sum_{\cdots}(\cdots) < \ell \Big) \vee \Big(-\sum_{\cdots}(\cdots) < -u \Big) \right). \tag{4.15}$$

If nested beliefs as defined in (4.14) contain negated belief operators, this can be expressed accordingly by replacing the conjunctive constraints on ℓ and u (resp. ℓ_j and u_j) with the corresponding disjunctive constraints (4.15) for negated atomic belief formulae.

Disjunctive Belief Formulae With the above inequalities, the required constraints for a disjunctive formula $\mathscr{B} = \mathscr{B}_1 \vee \mathscr{B}_2$ can easily be expressed as an additional disjunction of inequalities. Let C_1 and C_2 be the sets of inequalities to express satisfiability of $\mathscr{B}_1$ and $\mathscr{B}_2$ according to (4.12)–(4.15), respectively. Then, the constraints for $\mathscr{B}$ can be expressed as

$$C_1 \vee C_2 \tag{4.16}$$

Example 4.4 (Trains continued)**.** In Example 4.1, a set of belief formulae $\mathfrak{B}$ has been given for the train example. To illustrate the extraction of linear constraints from this set, we continue to use the set of threads depicted in Figure 3.1 with a minor modification: to reflect the model specified in $\mathfrak{B}$ of Example 4.1, we assume that the predicate *punct*(*train*) is explicitly encoded in the respective threads. Moreover, for the sake of the example we assume that the prior probabilistic interpretations are yet unknown. We use $x_1, ..., x_9$ to denote these unknown probabilities. Note that for our example, we are only dealing with prior beliefs, i.e., we only have one equivalence class $\mathcal{C} = \hat{\mathcal{T}}$ and all scaling factors $a^n_{i,t'}$ are equal to one. This significantly eases the presentation of this example. Of course, in general we have to deal with both multiple equivalence classes and multiple varying scaling factors. As this highly increases complexity of the presentation, we refrain from giving explicit examples for these cases. The constraints from $\mathfrak{B}$ are extracted as follows:

- For belief $\mathscr{B}'_2 = B^{.81,.81}_{A,0}\big(r^{pfr}_0(at(T_1, C_A), punct(T_1))\big)$:

$$pfr(Th, at(T_1, C_A), punct(T_1), 0) = 1 \text{ for } Th \in \{Th_1, ..., Th_3\}$$
$$pfr(Th, at(T_1, C_A), punct(T_1), 0) = 0 \text{ for } Th \in \{Th_4, ..., Th_9\}$$

  and thus application of rule (4.13) yields the constraints

$$\begin{aligned} -x_1 - x_2 - x_3 &\leq -0.81 \\ x_1 + x_2 + x_3 &\leq 0.81 \end{aligned}$$

  In this special case where $\ell = u$, we can simplify this constraint to

$$x_1 + x_2 + x_3 = 0.81$$

  Since all of the rules exhibit this property, we slightly deviate from (4.13) and only give the equivalent equality constraints for subsequent rules in order to simplify the presentation.

  Accordingly, for belief $\mathscr{B}''_2 = B^{.81,.81}_{A,0}\big(r^{pfr}_0(at(T_2, C_C), punct(T_2))\big)$ we obtain:

$$pfr(Th, at(T_2, C_C), punct(T_2), 0) = 1 \text{ for } Th \in \{Th_1, Th_4, Th_5\}$$
$$pfr(Th, at(T_2, C_C), punct(T_2), 0) = 0 \text{ for } Th \in \{Th_2, Th_3, Th_6..., Th_9\}$$

  with the corresponding constraints

$$x_1 + x_4 + x_5 = 0.81$$

- For belief $\mathscr{B}_6 = B^{.93,.93}_{A,0}\big(r^{pfr}_2(Obs_{\{A\}}(\neg punct(train)), Obs_{\{AB\}}(\neg punct(train)))\big)$:

  for $Th \in \{Th_1, Th_3, Th_5, Th_9\}$:
  $\quad pfr(Th, \neg punct(train), Obs_{\{AB\}}(\neg punct(train)), 2) = 1,$
  for $Th \in \{Th_2, Th_4, Th_6\}$:
  $\quad pfr(Th, \neg punct(train), Obs_{\{AB\}}(\neg punct(train)), 2) = 0,$
  for $Th \in \{Th_7, Th_8\}$:
  $\quad pfr(Th, \neg punct(train), Obs_{\{AB\}}(\neg punct(train)), 2) = 0.5$

  and thus application of rule (4.13) yields the constraint

  $$x_1 + x_3 + x_5 + 0.5 \cdot x_7 + 0.5 \cdot x_8 + x_9 = 0.93$$

- For the remaining beliefs, the respective belief objects are satisfied in every thread and thus we only obtain the redundant constraints

  $$\sum_{k=1}^{9} x_k = 1.$$

  One can easily verify that prior probabilistic interpretation given in Example 3.6, i.e.,

  $$x = \begin{pmatrix} 0.7 & 0.02 & 0.09 & 0.02 & 0.09 & 0.01 & 0.02 & 0.02 & 0.03 \end{pmatrix}$$

  indeed is a solution with respect to the above constraints. Of course, for the given example, this solution was expected, as $\mathfrak{B}$ was defined such that it exactly reflects the situation described in the examples from the previous chapter.

4.3.3. Transformation into a Disjunctive Program

For every belief formula $\mathscr{B} \in \mathfrak{B}$, the above extractions of linear constraints yield a set of inequalities of the form

$$a_{i,1}x_1 \;+\; a_{i,2}x_2 \;+\; \ldots \;+\; a_{i,m}x_m \;\leq\; b_i, \tag{4.17}$$

with x_j representing the unknown prior probabilities of threads $Th_1, ..., Th_m$, the coefficients $a_{i,j}$ set to the respective values of $a^n_{i,t'}$ if they contribute to this constraint and set to zero otherwise, and the value b_1 set to the respective limit obtained through ℓ or u.

As Corollary 4.7 shows, every belief formula $\mathscr{B} \in \mathfrak{B}$ yields a disjunctive set of inequality constraints, i.e., every belief formula $\mathscr{B}$ introduces branches in the set of linear constraints. By collecting all inequalities of the form (4.17) that constrain a single branch, we can express the constraints in matrix form:

$$Ax \leq b, \tag{4.18}$$

with

$$A = \begin{pmatrix} a_{1,1} & \cdots & a_{1,m} \\ \vdots & \ddots & \vdots \\ a_{n,1} & \cdots & a_{n,m} \end{pmatrix}, \; x = \begin{pmatrix} x_1 \\ \cdots \\ x_m \end{pmatrix}, \text{ and } b = \begin{pmatrix} b_1 \\ \cdots \\ b_m \end{pmatrix}$$

This form of representation has a close connection to linear programming (LP). Linear programming (e.g., [Mur83]) is a solution method to optimization problems where some linear function of a set of continuous variables x_k is to be optimized with respect to a given set of linear constraints. While the task of satisfiability checking does not require any optimization and thus actually solving a linear program is not required for this work, we will exploit similarities between our sets of linear constraints and LP in order to show how the satisfiability problem can be solved.

The standard form of an LP problem [Mur83] gives a set of constraints exactly in the form of (4.18). Every solution x that satisfies these constraints is called *feasible* and the entire solution space for (4.18) is called *feasible region.* Thus, checking whether a set of belief formulae $\mathfrak{B}$ is satisfiable is equivalent to checking whether a corresponding LP problem has a non-empty feasible region. For standard LP problems with constraints of the form (4.18), the feasible region is a convex polytope, which allows performing this check with little computational effort [GJ79].

Unfortunately, extracting linear constraints from a set of belief formulae $\mathfrak{B}$ as described in Section 4.3.2 does not yield a single set of constraints in the form of (4.18), but instead a disjunction of different sets of constraints. This gives rise to the representation of the satisfiability checking problem as a disjunctive program (DP) [Bal98]:

Corollary 4.8 (Satisfiability Checking as a Disjunctive Program)**.** *Let $\mathfrak{B}$ be a set of belief formulae, let $\hat{\mathcal{T}}$ be a set of threads and let D be the set of all disjunctive branches d of linear constraints extracted from $\mathfrak{B}$ and $\hat{\mathcal{T}}$ according to the extraction rules (4.12)-(4.16). Then, the satisfiability checking problem can be formulated as a disjunctive program [Bal98]:*

$$\bigvee_{d \in D} A^d x \leq b^d \tag{4.19}$$

$\mathfrak{B}$ is satisfiable with respect to $\hat{\mathcal{T}}$, denoted by $sat(\mathfrak{B}, \hat{\mathcal{T}})$, if (4.19) has a solution.

A disjunctive program is called bounded, if the range of every variable x_k is restricted through lower and upper bounds. Since we will rely on the bounded property subsequently, we state the following result:

Lemma 4.9 (Satisfiability Checking as a Bounded DP). *Let $\mathfrak{B}$ be a set of belief formulae and $\hat{\mathcal{T}}$ be a set of threads. Checking satisfiability of $\mathfrak{B}$ with respect to $\hat{\mathcal{T}}$ can be represented as a bounded disjunctive program.*

Proof. This is a straightforward result: Corollary 4.8 shows that satisfiability checking for PDT Logic can be represented as a disjunctive program in the form of (4.19). Since every variable x_k in (4.19) represents a probability value, all x_k are naturally bounded by $0 \leq x_k \leq 1$. □

In a disjunctive program, the feasible region cannot be guaranteed to be convex anymore, nor can it be guaranteed that the solution space even represents a connected region. This significantly increases the complexity of determining whether a nonempty solution space exists. To analyze this problem in more detail and to show connections to established solution approaches, we will discuss in the next section how a disjunctive program in the form (4.19) can be further transformed.

4.3.4. Transformation into a 0-1 Mixed Integer Linear Program

The concept of linear programs with continuous variables x_k subject to linear constraints of the form (4.17) can be extended to so-called mixed integer linear programs (MILPs) [Sch86]. Opposed to standard linear programming, for MILPs it is not required that all variables x_k have a continuous domain. Instead, MILPs can use a mix of both continuous and integer variables. There are several equivalent ways of representing a MILP, we adopt the representation from [FGL05], which specifies the constraints of a MILP as

$$\begin{aligned} Ax &\leq b \\ x_j &\text{ integer} \qquad \forall j \in I \end{aligned}$$

with an index set I indicating which of the variables x_j are integer variables. A special case of MILPs are 0-1 mixed integer linear programs [Wil09], where the integer variables x_j are restricted to binary values:

$$\begin{aligned} Ax &\leq b \\ x_j &\in \{0,1\} \qquad \forall j \in I \end{aligned} \tag{4.20}$$

By augmenting the set of variables x with binary switching variables x_j for every possible disjunction, it is possible to represent disjunctive programs in the form of

(4.19) as 0-1 MILPs in the form of (4.20) [Bal85]. This leads to a central result for satisfiability checking in PDT Logic:

Theorem 4.10 (Satisfiability Checking as 0-1 MILP)**.** *Let $\mathfrak{B}$ be a set of belief formulae and $\hat{\mathcal{T}}$ be a set of threads. The problem of checking satisfiability of $\mathfrak{B}$ with respect to $\hat{\mathcal{T}}$ can be transformed into a corresponding 0-1 mixed integer linear program M so that $\mathfrak{B}$ is satisfiable with respect to $\hat{\mathcal{T}}$ iff M has a feasible solution.*

Proof. Lemma 4.9 shows that satisfiability checking for PDT Logic can be represented as a bounded disjunctive program, such that a set of belief formulae $\mathfrak{B}$ is satisfiable iff the corresponding bounded disjunctive program has a feasible solution. The proof of Theorem 4.4 in [Bal85] shows that every bounded disjunctive program can be equivalently represented as a 0-1 mixed integer program M. Consequently, satisfiability checking for PDT Logic is equivalent to checking whether M has a feasible solution. □

We can leverage Theorem 4.10 to obtain complexity results for the satisfiability problem in PDT Logic:

Theorem 4.11 (Complexity of PDT SAT w.r.t. a given set of threads)**.** *Checking satisfiability of a set of PDT Logic belief formulae $\mathfrak{B}$ with respect to a given set of threads $\hat{\mathcal{T}}$ is NP-complete.*

Proof. It is generally known that checking whether a bounded 0-1 mixed integer linear program has a feasible solution is NP-complete (cf. [Bie96]). As Theorem 4.10 shows that satisfiability checking in PDT Logic with respect to a given set of threads $\hat{\mathcal{T}}$ can be reformulated as a 0-1 MILP with bounded variables x_k (cf. Lemma 4.9), it follows that satisfiability checking for a set of belief formulae $\mathfrak{B}$ with respect to a given set of threads $\hat{\mathcal{T}}$ is in NP.

Arbitrary propositional formulae F (cf. Definition 3.2) can be expressed in PDT Logic by using them as a belief object for a strict prior belief $B_{i,0}^{1,1}(F)$. Since it is well known that the boolean satisfiability problem (SAT) is NP-complete [Coo71], it follows that any problem in NP can be transformed to a satisfiability checking problem in PDT Logic. Hence, the satisfiability checking problem in PDT Logic is NP-hard and consequently NP-complete. □

The NP-completeness result shows that the problem is in NP and therefore we immediately obtain another important property of the satisfiability problem in PDT Logic:

Corollary 4.12 (Decidability of PDT SAT)**.** *Checking satisfiability of a set of PDT Logic belief formulae $\mathfrak{B}$ is decidable.*

MILPs have been subject to extensive research for decades, and thus an ample variety of solving methods has been proposed (e.g., [BCC93], [BCC96], [BP02], to name some of the most notable work on MILP solving, and especially [FGL05] and [BFL07] to find feasible solutions of MILPs). This research gave rise to various efficient implementations of MILP solvers, both commercial (e.g., [ILO], [Gur]) and non-profit products (e.g., [Gnu], [Com]). For a given set of threads, PDT Logic satisfiability checking can be reformulated as a 0-1 MILP problem, and thus any of these state-of-the-art MILP solvers can be exploited for relatively fast satisfiability checks for most instances of PDT Logic belief formulae $\mathfrak{B}$ with respect to a given set of threads $\hat{\mathcal{T}}$.

The results from this section show how satisfiability of a set of PDT Logic belief formulae $\mathfrak{B}$ can be decided with respect to a given set of threads, even if no specific prior probability assignment is specified. As the overall goal of this chapter is the design of a decision procedure that requires only a set of belief formulae $\mathfrak{B}$ as an input, we continue the discussion of satisfiability testing with the development of a method to automatically construct a set of threads representing the background knowledge specified in $\mathfrak{B}$.

4.4. Prior Constraints on Possible Threads

To determine whether the set of belief formulae $\mathfrak{B}$ is satisfiable, we need to obtain a set of possible threads that reflects the background knowledge specified in $\mathfrak{B}$. In this section, we describe how we can identify certain constraints on the set of possible threads $\hat{\mathcal{T}}$ prior to actually starting to generate threads that represent the information specified in $\mathfrak{B}$. To identify such prior constraints, we discuss different properties of the belief formulae contained in $\mathfrak{B}$. Using these properties, we can create a taxonomy of belief formulae depending on the respective impact on the set of possible threads $\hat{\mathcal{T}}$. Beliefs with certain properties can then be used to constrain the search space for sets of possible threads prior to actually search for these sets. After discussing prior constraints in this section, we use these results in Section 4.5 to develop a decision procedure for PDT Logic that requires neither a specification of probabilities nor a specification of possible threads.

4.4.1. A Taxonomy of Belief Formulae

The set of belief formulae $\mathfrak{B}$ may contain beliefs with various features that will have different impacts on the sets of admissible worlds at specific time points t. We will discuss these features below and show how they yield a taxonomy of belief formulae. This taxonomy allows for the classification of beliefs into three different types with respect to their impact on the sets of admissible worlds. In particular, we can identify beliefs that are independent of any specific probability assignment and of any Kripke relations $\mathcal{K}_i$. This classification is for technical purposes: beliefs that depend neither on specific probability assignments nor on specific Kripke relations can be used to derive initial constraints on the sets of possible worlds at some or all time points $t \in t_{max}$. We use $\hat{\Omega}_{\mathfrak{B}}$ to denote the set of all worlds admissible with respect to a set of belief formulae $\mathfrak{B}$, and we use $\hat{\Omega}_{\mathfrak{B}}(t)$ to denote the set of admissible worlds with respect to a set of belief formulae $\mathfrak{B}$ at time t.

Recall that there are three different kinds of beliefs: beliefs in facts, beliefs in rules, and beliefs in beliefs. As before, we differentiate between *prior* beliefs that hold at time point $t = 0$ (and are therefore commonly known among all agents) and *posterior* beliefs that hold at time points $t > 0$.

We can further distinguish beliefs in rules $B_{i,t'}^{\ell,u}(r_{\Delta t}^{pfr}(F, G))$ with respect to Δt: we call $r_{\Delta t}^{pfr}(F, G)$ a *static rule* if $\Delta t = 0$ and we call $r_{\Delta t}^{pfr}(F, G)$ a *dynamic rule* if $\Delta t > 0$. Accordingly, we can separate beliefs in rules into beliefs in static rules and beliefs in dynamic rules, respectively. These beliefs differ with respect to their temporal impact: a static rule will constrain the possible worlds instantaneously, i.e., $r_0^{pfr}(F, G)$ states that there can be no world ω such that both $\omega \models F$ and $\omega \not\models G$ hold. A dynamic rule on the other hand requires that whenever a world ω with $\omega \models F$ occurs, there must be another world ω' with $\omega' \models G$ after Δt time steps.

Finally, we can classify beliefs with respect to their probabilistic quantifications: we call a belief $B_{i,t'}^{\ell,u}(\varphi)$ *strict*, if both $\ell = u = 0$ or $\ell = u = 1$. For the sake of simplicity, in the following we assume without loss of generality that strict beliefs are always represented with $\ell = u = 1$. Any strict belief $B_{i,t}^{0,0}(\varphi)$ can easily be rewritten as $B_{i,t}^{1,1}(\neg\varphi)$.[1] We call a belief *trivial* if $\ell = 0$ and $u = 1$. Obviously, these beliefs are trivially satisfied by any arbitrary interpretation, thus they do not impact satisfiability checking results at all and therefore can be removed from $\mathfrak{B}$.

Remark 4.2. From the definition of the belief semantics (Definition 3.11) it follows for the special case of strict beliefs $B_{i,t}^{1,1}(\varphi)$ that (i) agent i considers the occurrence of the belief object's complement $\neg\varphi$ as impossible and (ii) that this occurrence is indeed impossible. Thus, strict beliefs comply with the common definitions of knowledge as *justified true belief* and *belief that is stable with respect to the truth* (cf. e.g., [SLB09, page 433]). Consequently, we could also refer to a strict belief as knowledge and equivalently use the established knowledge operator $K_i(\varphi)$ instead of $B_i^{1,1}(\varphi)$.

Remark 4.3. Note that the concept of strict beliefs only applies to positive beliefs

[1]If the belief object φ is a temporal rule $r_{\Delta t}^{\mathsf{fr}}(F, G)$, we represent $\neg\varphi$ as $r_{\Delta t}^{\mathsf{fr}}(F, \neg G)$. This is possible because we do not need to consider frequency functions that correspond to axioms FF1-FF3 from Definition 3.10 and only use point frequency functions *pfr*. If other frequency functions are used, their negations need to be defined accordingly.

$B^{1,1}_{i,t}(\varphi)$. For the negation of such a belief, $\neg B^{1,1}_{i,t}(\varphi)$, it follows from Definition 3.16 that there is at least one thread that does not satisfy the belief object φ, which in turn implies $\ell < 1$. Consequently, these beliefs $\neg B^{1,1}_{i,t}(\varphi)$ are considered as non-strict in the following discussion.

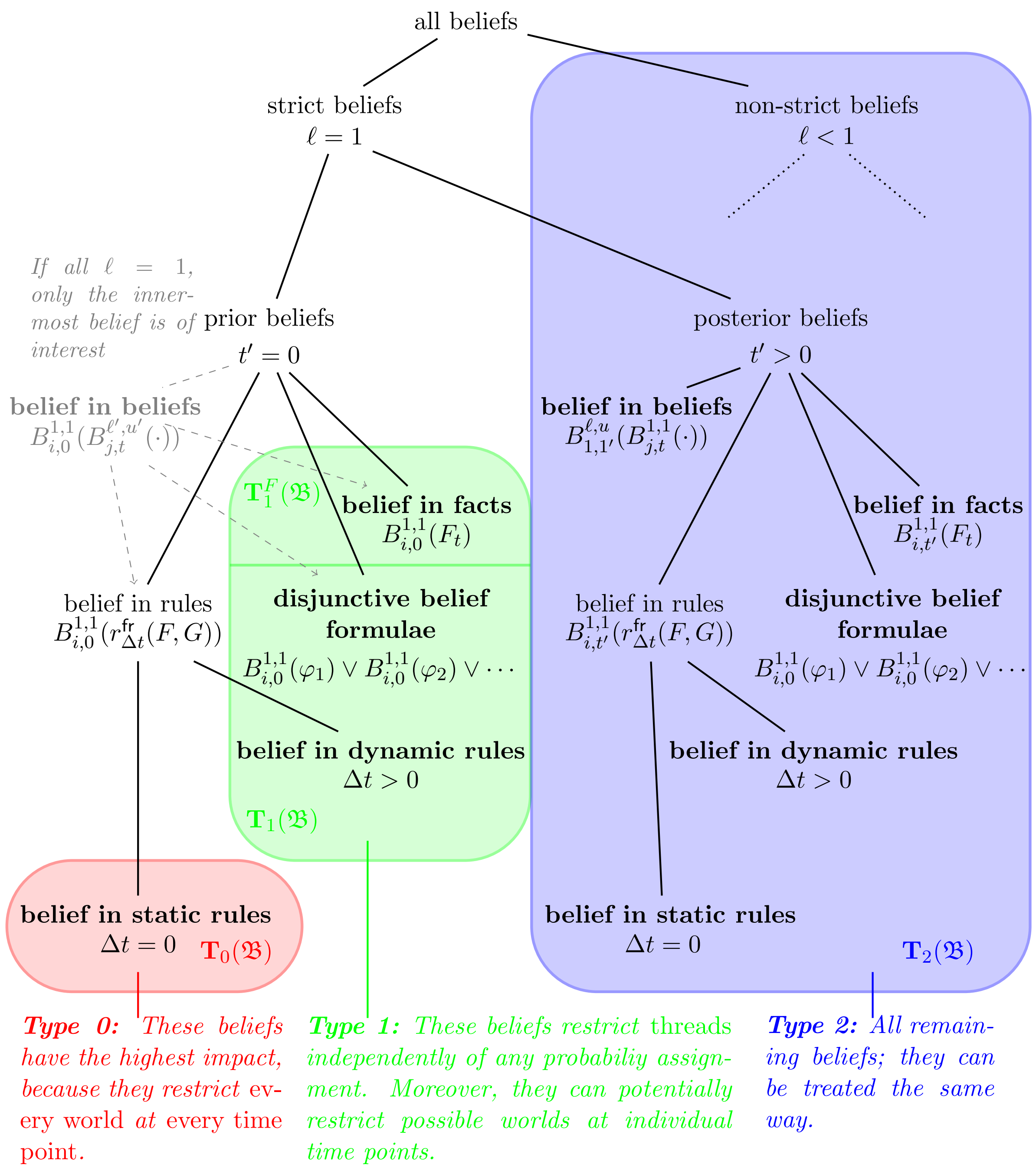
Using these features, we can create a taxonomy of beliefs as depicted in Figure 4.1 to identify prior constraints on the set of possible threads. This taxonomy is obtained by successively distinguishing between strict and non-strict, prior and posterior beliefs, between beliefs in facts, rules and nested beliefs, and finally between beliefs in static and dynamic rules. Nested beliefs are only considered as strict (prior) beliefs, if all involved beliefs are strict (prior), otherwise they are considered as non-strict (posterior). If a nested belief is actually strict and prior, we can unnest this belief and consider only the innermost belief expression: since prior beliefs are commonly known and therefore identical for all agents $i \in \mathcal{A}_{\mathfrak{B}}$, it is evident that for any strict belief of any agent i, all other agents know that agent i has this strict belief. Consequently, strict prior beliefs can be nested to an arbitrary depth without introducing any further constraints: they are satisfied exactly if the innermost belief is satisfied. Thus, we do not need to consider nested strict prior beliefs explicitly.

This taxonomy gives rise to three different types of belief formulae with respect to their impact on the sets of admissible worlds:

Definition 4.3 (Belief formula typification)**.** A set of belief formulae $\mathfrak{B}$ can be categorized into three different types of beliefs:

- **Type 0:** These are beliefs that restrict the set of admissible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ at every time point $t \in \tau$. Thus, type 0 beliefs have the highest impact because they can be exploited to prune the set of admissible worlds $\hat{\Omega}_{\mathfrak{B}}$ *globally.* An evaluation of these beliefs relies neither on a specific probability assignment nor on any given Kripke structures $\mathcal{K}_i$.
- **Type 1:** These are beliefs that restrict sequences of possible worlds. Moreover, they can potentially restrict the sets of admissible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ at specific time points. Thus, type 1 beliefs have less impact than type 0 beliefs because they can only be exploited to prune the sets of admissible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ *locally.* Again, an evaluation of these beliefs relies neither on a specific probability assignment nor on any given Kripke structures $\mathcal{K}_i$.
- **Type 2:** This type encompasses all remaining beliefs in $\mathfrak{B}$ that are neither type 0 nor type 1 beliefs. These beliefs are situation-specific and cannot be used to prune the sets of admissible worlds a priori. Satisfiability of these beliefs depends on a suitable probability assignment or on the evaluation of Kripke structures in the respective threads.

We use $\mathbf{T}_k(\mathfrak{B})$ to denote the set of type k beliefs from $\mathfrak{B}$.


all beliefs
strict beliefs $\ell = 1$
non-strict beliefs $\ell < 1$
If all $\ell = 1$, only the innermost belief is of interest
prior beliefs $t' = 0$
posterior beliefs $t' > 0$
belief in beliefs $B^{1,1}_{i,0}(B^{\ell',u'}_{j,t}(\cdot))$
belief in beliefs $B^{\ell,u}_{1,1'}(B^{1,1}_{j,t}(\cdot))$
$\mathbf{T}^F_1(\mathfrak{B})$
belief in facts $B^{1,1}_{i,0}(F_t)$
belief in facts $B^{1,1}_{i,t'}(F_t)$
belief in rules $B^{1,1}_{i,0}(r^{\mathsf{fr}}_{\Delta t}(F,G))$
belief in rules $B^{1,1}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(F,G))$
disjunctive belief formulae $B^{1,1}_{i,0}(\varphi_1) \vee B^{1,1}_{i,0}(\varphi_2) \vee \cdots$
disjunctive belief formulae $B^{1,1}_{i,0}(\varphi_1) \vee B^{1,1}_{i,0}(\varphi_2) \vee \cdots$
belief in dynamic rules $\Delta t > 0$
belief in dynamic rules $\Delta t > 0$
$\mathbf{T}_1(\mathfrak{B})$
belief in static rules $\Delta t = 0$
$\mathbf{T}_0(\mathfrak{B})$
belief in static rules $\Delta t = 0$
$\mathbf{T}_2(\mathfrak{B})$

Type 0: *These beliefs have the highest impact, because they restrict every world at every time point.*

Type 1: *These beliefs restrict threads independently of any probabiliy assignment. Moreover, they can potentially restrict possible worlds at individual time points.*

Type 2: *All remaining beliefs; they can be treated the same way.*

Figure 4.1.: Taxonomy of belief formulae

The main goal of this belief formula taxonomy is to identify constraints on possible worlds ω and possible threads Th that can be evaluated prior to searching for a suitable probability assignment, namely by using the belief formulae in $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$ to prune the search space of possible sets of threads $\hat{\mathcal{T}}$ that may show satisfiability of $\mathfrak{B}$. It should be noted that the existence of a thread $Th \in \hat{\mathcal{T}}$ violating a belief from $\mathbf{T}_0(\mathfrak{B})$ or $\mathbf{T}_1(\mathfrak{B})$ technically does not preclude satisfiability of $\mathfrak{B}$ with respect to $\hat{\mathcal{T}}$, as there is a special case of a suitable probability assignment: If there is a thread $Th \in \hat{\mathcal{T}}$ such that some belief $\mathscr{B} \in \mathbf{T}_0(\mathfrak{B})$ or $\mathscr{B} \in \mathbf{T}_1(\mathfrak{B})$ is not satisfied, there could still be suitable probability assignments $\mathcal{I}(\hat{\mathcal{T}})$ such that $sat(\mathfrak{B}, \hat{\mathcal{T}})$ holds iff $\mathcal{I}(Th) = 0$. The effect of excluding such a thread Th from $\hat{\mathcal{T}}$ or assigning a prior probability $\mathcal{I}(Th)$ of zero is the same (cf. Remark 3.3), i.e., the respective thread is marked as impossible. Since we aim at reducing both the search space of possible threads and the input to the satisfiability check $sat(\mathfrak{B}, \hat{\mathcal{T}})$, we exploit belief formulae in $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$ to exclude impossible threads prior to searching for suitable probability assignments.

Type 0 belief formulae As depicted in Figure 4.1, the set of type 0 belief formulae is formed by formulae with strict prior beliefs in static rules $B^{1,1}_{i,0}(r^{pfr}_0(F,G))$ from $\mathfrak{B}$. Since prior beliefs represent the background knowledge and since it follows from the definition of strict beliefs that they cannot be violated in any world, it is clear that the rule $r^{pfr}_0(F,G)$ has to be always satisfied. As this is a static rule, it has to be satisfied in *every world* $\omega \in \hat{\Omega}_{\mathfrak{B}}$. We define the set of type 0 beliefs as

$$\mathbf{T}_0(\mathfrak{B}) = \{\mathscr{B} \in \mathfrak{B} :\ \mathscr{B} = B^{1,1}_{i,0}(r^{pfr}_0(F,G))\} \tag{4.21}$$

with arbitrary formulae F and G.

Type 1 belief formulae The set of type 1 beliefs contains all strict prior beliefs that are not in the set $\mathbf{T}_0(\mathfrak{B})$. The contributions of this set $\mathbf{T}_1(\mathfrak{B})$ are twofold: As $\mathbf{T}_1(\mathfrak{B})$ only comprises strict prior beliefs, *every thread* in a potential set of threads $\hat{\mathcal{T}}$ has to satisfy all beliefs $\mathscr{B} \in \mathbf{T}_1(\mathfrak{B})$. Moreover, constraints from $\mathbf{T}_1(\mathfrak{B})$ may constrain the sets of worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ at individual time points $t \in \tau$ regardless of any specific thread. According to Figure 4.1, we define the set of type 1 beliefs as

$$\begin{aligned}\mathbf{T}_1(\mathfrak{B}) = \big\{\mathscr{B} \in \mathfrak{B} : \big(\ & \mathscr{B} = B^{1,1}_{i,0}(F_t)\\ & \vee \mathscr{B} = (B^{1,1}_{i,0}(r^{pfr}_{\Delta t}(F,G)) \wedge \Delta t > 0)\\ & \vee \mathscr{B} = (B^{1,1}_{i,0}(\varphi_1) \vee B^{1,1}_{i,0}(\varphi_2) \vee \cdots)\big)\big\}\end{aligned} \tag{4.22}$$

For a potential set of possible threads $\hat{\mathcal{T}}$, the beliefs specified in this set $\mathbf{T}_1(\mathfrak{B})$ have to be satisfied by every thread $Th \in \hat{\mathcal{T}}$. Note that satisfiability of beliefs in dynamic rules and disjunctive belief formulae generally depends on worlds ω at multiple time points and thus satisfiability of $\mathbf{T}_1(\mathfrak{B})$ cannot be ensured by only constraining sets of worlds at single time points. However, by analyzing strict prior beliefs in facts and their potential interplay with dynamic rules we can derive constraints for the sets of worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ at specific time points $t \in \tau$ as follows.

Strict prior beliefs in facts $\mathscr{B} = B_{i,0}^{1,1}(F_t)$ restrict the set of admissible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ at time t by enforcing that F holds at every world $\omega \in \hat{\Omega}_{\mathfrak{B}}(t)$. In the following, we use $\mathbf{T}_1^F(\mathfrak{B})$ to denote such strict prior beliefs in facts F at certain time points t. Moreover, we use $\mathfrak{B} \models F_t$ as a shorthand for $B_{i,0}^{1,1}(F_t) \in \mathfrak{B}$ to denote that $\mathfrak{B}$ enforces F at time t.

Through interplay with existing constraints on sets possible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ at individual time points t, strict beliefs in dynamic rules can yield additional constraints: For a belief formula $\mathscr{B} = B_{i,0}^{1,1}(r_{\Delta t}^{pfr}(F,G)), \Delta t > 0$, additional constraints might be derived, depending on the type of belief in the respective rule's premise F: if $(\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1^F(\mathfrak{B})) \models F_t$ is given, we can extract a strict prior belief in a fact $\mathscr{B}' = B_{i,0}^{1,1}(G_{t+\Delta t})$, which then again restricts the set of possible worlds at time point $t + \Delta t$ and is therefore added to $\mathbf{T}_1^F(\mathfrak{B})$.

Since dynamic rules can be considered as temporal implications (cf. Definition 3.10 from Chapter 3), these rules can also be applied backwards to obtain additional constraints: If a belief formula $\mathscr{B} = B_{i,0}^{1,1}(r_{\Delta t}^{pfr}(F,G)), \Delta t > 0$ is given and the rule's negated conclusion $\neg G$ is already enforced at some time point t (i.e., $(\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1^F(\mathfrak{B})) \models \neg G_t$), the rule's premise F cannot be satisfied at time $t - \Delta t$. Thus, we can add the belief $\mathscr{B}' = B_{i,0}^{1,1}(\neg F_{t-\Delta t})$ as an additional constraint to $\mathbf{T}_1^F(\mathfrak{B})$.

Extending the set of type 1 beliefs through dynamic rules may lead to a chained extension: if we have a belief in a dynamic rule $B_{i,0}^{1,1}(r_{\Delta t}^{pfr}(F,G))$ and a corresponding belief $B_{i,0}^{1,1}(F_t) \in \mathbf{T}_1^F(\mathfrak{B})$, this will lead to the additional belief $B_{i,0}^{1,1}(G_{t+\Delta t}) \in \mathbf{T}_1^F(\mathfrak{B})$, which in turn might trigger another dynamic rule $B_{i,0}^{1,1}(r_{\Delta t}^{pfr}(G,G'))$. Analogously, any additional belief in $\mathbf{T}_1^F(\mathfrak{B})$ could also trigger further backward rule applications.

To capture all constraints that emerge from forward and backward chaining of strict dynamic rules, we define the set $\mathbf{T}_1^F(\mathfrak{B})$ as the following fix-point set:[2]

$$\begin{aligned}\mathbf{T}_1^F(\mathfrak{B}) = \quad & \{B_{i,0}^{1,1}(F_t) \in \mathfrak{B}\} \\ & \cup \{B_{i,0}^{1,1}(G_{t+\Delta t}) : \ \big(\Delta t > 0 \ \wedge \ B_{i,0}^{1,1}(r_{\Delta t}^{pfr}(F,G)) \in \mathfrak{B} \\ & \qquad\qquad\qquad\qquad \wedge \ \big((\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1(\mathfrak{B})) \models \ \ F_t\big)\big)\} \\ & \cup \{B_{i,0}^{1,1}(\neg F_{t-\Delta t}) : \ \big(\Delta t > 0 \ \wedge \ B_{i,0}^{1,1}(r_{\Delta t}^{pfr}(F,G)) \in \mathfrak{B} \\ & \qquad\qquad\qquad\qquad \wedge \ \big((\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1(\mathfrak{B})) \models \neg G_t\big)\big)\}\end{aligned} \tag{4.23}$$

After having determined all constraints on individual time points, we can reduce this set $\mathbf{T}_1^F(\mathfrak{B})$ such that it contains at most one belief $B_{i,0}^{1,1}(F_t)$ for every time point t. If $\mathbf{T}_1^F(\mathfrak{B})$ contains multiple beliefs $B_{i,0}^{1,1}(F_t)$, $B_{i,0}^{1,1}(G_t)$ regarding the same time point t, we can replace them by a joint belief $B_{i,0}^{1,1}(F'_t)$ with $F' = F \wedge G$. Note that this substitution uses Lemma 3.7 to merge different belief expressions into one expression with a conjunctive *belief object*. We still assume that belief formulae with conjunctions of *belief operators* are separated into atomic belief formulae.

Type 2 belief formulae The set of type 2 belief formulae consists of all beliefs in $\mathfrak{B}$ that are neither type 0 nor type 1 beliefs. Thus we define this set as

$$\mathbf{T}_2(\mathfrak{B}) = (\mathfrak{B} \setminus \mathbf{T}_0(\mathfrak{B})) \setminus \mathbf{T}_1(\mathfrak{B}) \tag{4.24}$$

Example 4.5 (Trains continued)**.** Continuing with the set of belief formulae $\mathfrak{B}$ from Example 4.1 and assuming that conjunctive formulae $\mathscr{B} = \mathscr{B}' \wedge \mathscr{B}''$ are treated as individual formulae $\mathscr{B}'$ and $\mathscr{B}''$, we obtain the following sets of typed

[2]For this representation, we have only considered the influence of temporal rules for the set $\mathbf{T}_1^F(\mathfrak{B})$. In principle, information from disjunctive formulae $\mathscr{B} = B_{i,0}^{1,1}(\varphi_1) \vee \cdots \vee B_{i,0}^{1,1}(\varphi_n)$ in $\mathbf{T}_1(\mathfrak{B})$ could yield additional constraints on the sets $\hat{\Omega}_{\mathfrak{B}}(t)$: If $\mathbf{T}_1^F(\mathfrak{B})$ enforces $n-1$ disjuncts in $\mathscr{B}$ to be false, the remaining disjunct must be satisfied. As the belief objects of the respective disjuncts might be dynamic rules again, a formal representation of this consideration would result in a rather intricate specification. Since we have to ensure that any potential thread satisfies all beliefs in $\mathbf{T}_1(\mathfrak{B})$ anyways, omitting disjunctive formulae in the construction of $\mathbf{T}_1^F(\mathfrak{B})$ does not impact satisfiability results. Yet an actual implementation of the described procedures could exploit this consideration to obtain additional pruning conditions in special cases.

belief formulae:

$$\mathbf{T}_0(\mathfrak{B}) = \big\{B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_0(\neg \mathit{punct}(\mathit{train}) \land \mathit{at}(\mathit{train},\mathit{city}), \mathit{Obs}_{\{A\}}(\neg \mathit{punct}(\mathit{train})))\big)\big\} \quad (\mathscr{B}_5)$$

$$\begin{aligned}
\mathbf{T}_1(\mathfrak{B}) = \big\{ & B^{1,1}_{A,0}\big(\mathit{at}(T_1,C_A)_1\big), && (\mathscr{B}'_1)\\
& B^{1,1}_{A,0}\big(\mathit{on}(A,T_1)_1\big), && (\mathscr{B}''_1)\\
& B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_3(\ \mathit{punct}(T_1) \land \mathit{at}(T_1,C_A),\ \mathit{at}(T_2,C_C) \land \mathit{on}(A,T_2))\big), && (\mathscr{B}'_3)\\
& B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_5(\neg \mathit{punct}(T_1) \land \mathit{at}(T_1,C_A),\ \mathit{at}(T_2,C_C) \land \mathit{on}(A,T_2))\big), && (\mathscr{B}''_3)\\
& B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_2(\ \mathit{punct}(T_2) \land \mathit{at}(T_2,C_C),\ \mathit{at}(T_2,C_B) \land \mathit{on}(A,T_2))\big) && (\mathscr{B}'_4)\\
& B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_3(\neg \mathit{punct}(T_2) \land \mathit{at}(T_2,C_C),\ \mathit{at}(T_2,C_B) \land \mathit{on}(A,T_2))\big)\big\}, && (\mathscr{B}''_4)
\end{aligned}$$

$$\begin{aligned}
\mathbf{T}^F_1(\mathfrak{B}) = \big\{ & B^{1,1}_{A,0}\big(\mathit{at}(T_1,C_A)_1\big), && (\mathscr{B}'_1)\\
& B^{1,1}_{A,0}\big(\mathit{on}(A,T_1)_1\big)\big\}, && (\mathscr{B}''_1)
\end{aligned}$$

$$\begin{aligned}
\mathbf{T}_2(\mathfrak{B}) &= \mathfrak{B} \setminus \mathbf{T}_0(\mathfrak{B}) \setminus \mathbf{T}_1(\mathfrak{B})\\
&= \{B^{.81,.81}_{A,0}\big(r^{\mathit{pfr}}_0(\mathit{at}(T_1,C_A), \mathit{punct}(T_1))\big), && (\mathscr{B}'_2)\\
&\quad\ B^{.81,.81}_{A,0}\big(r^{\mathit{pfr}}_0(\mathit{at}(T_2,C_C), \mathit{punct}(T_2))\big), && (\mathscr{B}''_2)\\
&\quad\ B^{.93,.93}_{A,0}\big(r^{\mathit{pfr}}_2(\mathit{Obs}_{\{A\}}(\neg \mathit{punct}(\mathit{train})), \mathit{Obs}_{\{AB\}}(\neg \mathit{punct}(\mathit{train})))\big)\} && (\mathscr{B}_6)
\end{aligned}$$

The taxonomy of belief formulae provides means to construct sets of admissible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ for every time point $t \in \tau$. Type 0 beliefs (i.e., beliefs with the highest impact) constrain the global set of possible worlds $\hat{\Omega}_{\mathfrak{B}}$. Certain beliefs of type 1—materialized in the set $\mathbf{T}^F_1(\mathfrak{B})$—can then give additional constraints for specific time points t, such that only subsets $\hat{\Omega}_{\mathfrak{B}}(t) \subseteq \hat{\Omega}_{\mathfrak{B}}$ need to be considered as possible worlds for time t. The sets $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$ together provide satisfiability conditions that are independent of any specific probability assignments. Then, only beliefs of type 2 need to be considered as probabilistic constraints to check whether $\mathfrak{B}$ can be satisfied with respect to $\hat{\mathcal{T}}$, i.e., the satisfiability problem $\mathit{sat}(\mathfrak{B}, \hat{\mathcal{T}})$ from the previous section can be reduced to $\mathit{sat}(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}})$, if unsatisfiability of $\mathfrak{B}$ has not yet been shown through constraints in $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$. Since the prior constraints define necessary conditions for any potential thread, they give rise to a definition of thread soundness with respect to a given set of belief formulae $\mathfrak{B}$:

Definition 4.4 (Thread soundness)**.** Let $\mathfrak{B}$ be a set of belief formulae, and let $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$ be the set of type 0 and type 1 belief formulae in this set, respectively.

Then, a thread Th is *sound* with respect to $\mathfrak{B}$ (denoted $snd(Th, \mathfrak{B})$) if it satisfies all belief formulae from $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$:

$$snd(Th, \mathfrak{B}) \equiv \forall \mathscr{B} \in (\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1(\mathfrak{B})) : Th \models \mathscr{B} \tag{4.25}$$

Accordingly, we use $snd(\hat{\mathcal{T}}, \mathfrak{B})$ to denote that all threads $Th \in \hat{\mathcal{T}}$ are sound.

Note that this definition only relies on strict prior beliefs and the soundness property can therefore be verified for every thread individually, without having to consider other threads or probability assignments. Thus, a simplified version of the model checking procedure from Section 4.1 can be used to verify soundness. The intuition behind this property is that we can verify it easily prior to checking $sat(\mathfrak{B}, \hat{\mathcal{T}})$ and can therefore obtain a reduced version of the satisfiability problem:

Theorem 4.13 (Reduced satisfiability checking)**.** *Let $\mathfrak{B}$ be a set of belief formulae, let $\mathbf{T}_2(\mathfrak{B})$ be the set of type 2 beliefs in $\mathfrak{B}$ according to (4.24), and let $\hat{\mathcal{T}}$ be a set of sound threads. Then, $\mathfrak{B}$ is* satisfiable *with respect to $\hat{\mathcal{T}}$ iff $\mathbf{T}_2(\mathfrak{B})$ is satisfiable with respect to $\hat{\mathcal{T}}$:*

$$sat(\mathfrak{B}, \hat{\mathcal{T}}) \wedge snd(\hat{\mathcal{T}}, \mathfrak{B}) \;\equiv\; sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}) \tag{4.26}$$

Proof. This follows directly from Definition 4.4: $snd(\hat{\mathcal{T}}, \mathfrak{B})$ is defined so that it satisfies all belief formulae in the sets $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$. Consequently, these sets resemble tautologies with respect to $\hat{\mathcal{T}}$ and therefore do not have any impact on the satisfiability checking properties. Thus, instead of checking $\mathfrak{B}$ for satisfiability, it suffices to check the set $(\mathfrak{B} \setminus \mathbf{T}_0(\mathfrak{B})) \setminus \mathbf{T}_1(\mathfrak{B})$, which is exactly the definition of $\mathbf{T}_2(\mathfrak{B})$. □

4.4.2. Constraining Possible Worlds at Individual Time Points

Using the classification of beliefs in $\mathfrak{B}$ into the three different types, we can now continue with constructing sets of possible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ for every time point $t \in \tau$. The main goal of this section is an identification of obvious pruning conditions for possible worlds at specific time points. Since we are in the process of searching for a set of possible threads that satisfies a set of belief formulae $\mathfrak{B}$, any constraints on the sets $\hat{\Omega}_{\mathfrak{B}}(t)$ have the potential to significantly reduce the later used search space. Thus, the results of this section highlight possible optimizations for an implementation of a PDT Logic sat solver. Even if the following constraints are not—or only partially—applied, the search for possible threads as described in subsequent Section 4.5 can be carried out, yet with a potentially larger search space.

Since the set of type 0 beliefs has to be satisfied in every admissible world, we can define the global set of admissible worlds $\hat{\Omega}_{\mathfrak{B}}$ as follows:

Definition 4.5 (Global set of admissible worlds)**.** Let $\mathfrak{B}$ be a set of belief formulae, with the corresponding sets of belief objects $\mathbf{F}_{\mathfrak{B}}$ and type 0 beliefs $\mathbf{T}_0(\mathfrak{B})$. Then, the set of admissible worlds $\hat{\Omega}_{\mathfrak{B}}$ w.r.t. $\mathfrak{B}$ is given as

$$\hat{\Omega}_{\mathfrak{B}} = \left\{ \omega \in \mathcal{B}^{\mathbf{F}_{\mathfrak{B}}} : \left(adm(\omega) \wedge \forall B_{i,0}^{1,1}(r_0^{pfr}(F,G)) \in \mathbf{T}_0(\mathfrak{B}) :\ \omega \models (\neg F \vee G) \right) \right\}. \quad (4.27)$$

Remark 4.4. This definition uses $adm(\omega)$ to ensure that all worlds $\omega \in \hat{\Omega}_{\mathfrak{B}}$ are admissible as defined in the external Definition 3.5. Alternatively, we could use the existing formalism to encode these admissibility conditions directly as strict prior beliefs in $\mathfrak{B}$: $B_{i,0}^{1,1}(r_0^{pfr}(Obs_{\mathcal{G}}(l), l))$ and $\forall \mathcal{G}' \subset \mathcal{G} :\ B_{i,0}^{1,1}(r_0^{pfr}(Obs_{\mathcal{G}}(l), Obs_{\mathcal{G}'}(l)))$ represent conditions 1 and 2 of Definition 3.5, respectively. However, since these conditions are independent of the respective problem being modeled, we do not include them in the problem-specific belief set $\mathfrak{B}$, but use them as external constraints.

Example 4.6 (Trains continued)**.** The global set of worlds $\hat{\Omega}_{\mathfrak{B}}$ admissible with respect to $\mathfrak{B}$ from Example 4.1 can be automatically constructed from all combinations of events from $\mathbf{F}_{\mathfrak{B}}$ shown in Example 4.2, given that these combinations are admissible with respect to Definition 3.5 and satisfy the type 0 beliefs in $\mathbf{T}_0(\mathfrak{B})$ from Example 4.4. We refrain from enumerating all of these worlds explicitly and instead describe which worlds are excluded from the Herbrand base $\mathcal{B}^{\mathbf{F}_{\mathfrak{B}}}$ of $\mathbf{F}_{\mathfrak{B}}$: From $\mathbf{F}_{\mathfrak{B}}$ it follows that the only possible shared observation between A and B is the fact that a train is not punctual $(Obs_{\{AB\}}(\neg punct(train)))$. In every possible world where this observation occurs, admissibility conditions require that both agents A and B observe that the respective train is not punctual and that the train is indeed not punctual. Furthermore, the beliefs in $\mathbf{T}_0(\mathfrak{B})$ require that there is a corresponding observation for A at every possible world where a train is not punctual (which incidentally also enforces admissibility conditions for these observations).

Next, we can build upon the set of globally admissible worlds $\hat{\Omega}_{\mathfrak{B}}$ and use the set of type 1 beliefs to further prune the set of admissible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ at individual time points t:

Definition 4.6 (Local sets of admissible worlds)**.** Let $\mathfrak{B}$ be a set of belief formulae with the corresponding sets of admissible worlds $\hat{\Omega}_{\mathfrak{B}}$, $\mathbf{T}_1^F(\mathfrak{B})$ be the set of materialized strict prior beliefs induced by $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$, and τ be a set of time points. Then, the set of admissible worlds $\hat{\Omega}_{\mathfrak{B}}(t)$ w.r.t. $\mathfrak{B}$ at time $t \in \tau$ is given as

$$\hat{\Omega}_{\mathfrak{B}}(t) = \left\{ \omega \in \hat{\Omega}_{\mathfrak{B}} : \left(\forall B_{i,0}^{1,1}(F_t) \in \mathbf{T}_1^F(\mathfrak{B}) :\ \omega \models F \right) \right\}. \quad (4.28)$$

Example 4.7 (Trains continued)**.** To obtain the scenario from the original Example 3.2, we assume $t_{max} = 9$. From the set $\mathbf{T}_1^F(\mathfrak{B})$ identified in Example 4.5, we can restrict the set of worlds at time 1 to

$$\hat{\Omega}_{\mathfrak{B}}(1) = \Big\{\omega \in \hat{\Omega}_{\mathfrak{B}} : \omega \models (at(T_1, C_A) \wedge on(A, T_1))\Big\}$$

For all other time points, there are no options for further restrictions, thus the respective local sets $\hat{\Omega}_{\mathfrak{B}}(t)$ of possible worlds for all time points $t \neq 1$ remain at $\hat{\Omega}_{\mathfrak{B}}$.

Using Definition 4.6, we can now formulate constraints for the set of sound threads $\hat{\mathcal{T}}$:

$$\forall Th \in \hat{\mathcal{T}},\ \forall t \in \tau : Th(t) \in \hat{\Omega}_{\mathfrak{B}}(t). \tag{4.29}$$

Note that this constraint provides a necessary but not sufficient condition for thread soundness. To illustrate this, consider Example 4.5 again: the set $\mathbf{T}_1^F(\mathfrak{B})$ requires that $\{at(T_1, C_A), on(A, T_1)\}$ holds at every possible world at time $t = 1$ and thus we can constrain $\hat{\Omega}_{\mathfrak{B}}(1)$ as shown in Example 4.7, because any thread violating this constraint is inherently unsound. On the other hand, a thread according to (4.29) may contain the fact, say $punct(T_1) \in Th(1)$, which—according to $\mathscr{B}_3'$—only yields a sound thread if $\{at(T_2, C_C), on(A, T_2)\} \subseteq Th(4)$ holds as well. Thus, (4.29) provides general constraints on the set of threads with respect to beliefs from $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1^F(\mathfrak{B})$, while additional beliefs from $\mathbf{T}_1(\mathfrak{B})$ can discard individual threads by catching any potential unsatisfiable interplay of possible worlds at different time points.

Of course, in general it is possible that the methods discussed so far result in special cases: for one thing, it is possible that $\mathfrak{B}$ induces a set $\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1^F(\mathfrak{B})$ of inconsistent beliefs, i.e., it will contain beliefs that contradict each other. Then, $\hat{\Omega}_{\mathfrak{B}}$ or $\hat{\Omega}_{\mathfrak{B}}(t)$ for some t will be empty. This precludes the creation of any set of threads $\hat{\mathcal{T}}$ such that $\mathcal{I}(\hat{\mathcal{T}}) \models \mathfrak{B}$. In this case, satisfiability checking can terminate immediately with a negative result. For another, it is possible that the above simplification process will result in an empty set $\mathbf{T}_2(\mathfrak{B})$. In this case, there are no probabilistic constraints that could impact satisfiability of $\mathfrak{B}$ and thus it is unnecessary to search for a suitable probability assignment. In this case, it needs to be checked whether any of the threads in compliance with (4.29) is sound according to Definition 4.4. If such a thread can be found, satisfiability checking can terminate immediately with a positive result, otherwise $\mathfrak{B}$ is unsatisfiable. Verifying soundness of a single thread can be done with a simplified version of the model checking procedure from Section 4.1 and is therefore in *PTIME* (cf. Corollary 4.3). However, as the number threads satisfying condition (4.29) can grow exponentially with the number of ground atoms and the number of time points, the problem of finding a sound thread is more complex:

Theorem 4.14 (Complexity of finding a sound thread). *Let $\mathfrak{B}$ be a set of belief formulae such that all included formulae are grounded. Deciding whether there exists a sound thread with respect to $\mathfrak{B}$, as defined in Definition 4.4, is NP-complete.*

Proof. According to Definition 4.4, a set is sound if it satisfies all formulae from the set $\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1(\mathfrak{B})$. By treating the belief objects' atoms F at all time points t as individual variables F_t, we can transform beliefs in facts and belief in rules from $\mathbf{T}_0(\mathfrak{B}) \cup \mathbf{T}_1(\mathfrak{B})$ into a boolean sat problem as follows:[3]

$$\begin{aligned} B_{i,0}^{1,1}(F_t) &\quad \Rightarrow F_t \\ B_{i,0}^{1,1}(r_{\Delta t}^{\mathit{pfr}}(F,G)) &\quad \Rightarrow \bigwedge_{t=0}^{t_{max}-\Delta t} (\neg F_t \vee G_{t+\Delta t}) \end{aligned}$$

Accordingly, disjunctive belief formulae can then be expressed through transforming every disjunct individually. This transformation requires at most t_{max} conjuncts for every belief operator and can therefore be performed in linear time. Since the boolean sat problem is known to be NP-complete [Coo71], it follows that searching for a sound thread with respect to $\mathfrak{B}$ is in NP.

NP-hardness of this problem has already been shown in the proof of Theorem 4.11 and consequently it follows that searching for a sound thread with respect to $\mathfrak{B}$ is NP-complete. □

It should be noted that this result analyzes the worst-case complexity of the problem, but in practice finding a sound thread is usually not dominated by this worst case. In most cases, a sound thread can be found easily by employing the principle of least effort: For belief in temporal rules $B_{i,0}^{1,1}(r_{\Delta t}^{\mathit{pfr}}(F,G))$, choosing worlds ω such that $\omega \models \neg F$ ensures that consequences of this rule do not have to be evaluated at other time points. Accordingly, for disjunctive rules a disjunct should be selected such that no temporal rule is triggered by this fact. Of course, this is only a heuristic that may not give a sound thread immediately for every input $\mathfrak{B}$, but it represents a feasible approach for most problems. We will illustrate this approach with an example subsequently.

In this work, we only consider ground formulae for PDT Logic. In general, the formalism as introduced in Chapter 3 allows the treatment of non-ground formulae as well. However, for non-ground formulae the complexity result from Theorem 4.14 does not hold, because transformation into a boolean sat problem is then exponential in the number of possible groundings. Finding a sound thread then requires the use of sophisticated grounding procedures, (e.g., [DPDPR09] and [FLP12]), which is beyond the scope of this work.

[3]This transformation is only defined for temporal rules with point frequency functions *pfr*. If other frequency functions are used, the transformation has to be adapted accordingly.

Now that sets of possible worlds are identified for every time point $t \in \tau$, we can proceed with creating sets of representative threads with respect to these constraints. The aim of the following discussion is the successive generation of a set of representative threads $\hat{\mathcal{T}}$ such that $sat(\mathfrak{B}, \hat{\mathcal{T}})$ can be decided.

4.5. Representative Threads

Using Definition 4.4 and constraint (4.29) gives rise to a potential definition of the set of possible threads $\hat{\mathcal{T}}$ by constructing all possible combinations of sound world sequences from $\hat{\Omega}_{\mathfrak{B}}(t)$ for all $t \in \tau$. However, this would still result in an unnecessarily large set of possible threads. Instead of constructing all of these threads explicitly, we will heuristically create *representative threads* that represent excerpts from the situations modeled by $\mathbf{T}_2(\mathfrak{B})$. This approach uses heuristics to successively expand the set of representative threads. As soon as a suitable set of threads (i.e., a model for $\mathfrak{B}$) is found, the decision procedure can terminate with a positive result. If a set of representative threads does not show satisfiability of $\mathfrak{B}$, additional threads are created until either a positive satisfiability result is obtained or all possible threads have been created. Consequently, the heuristic search for models constitutes a complete decision procedure for PDT Logic.

For the following discussion, we assume that the set $\mathbf{T}_2(\mathfrak{B})$ is nonempty, i.e., there are additional constraints that need to be satisfied by the generated set of threads. Otherwise, if the set $\mathbf{T}_2(\mathfrak{B})$ was empty, satisfiability could already be determined by checking whether a sound thread with respect to $\mathfrak{B}$ exists, as discussed in the previous section and there would be no need to generate any specific set of threads.

For all beliefs in facts $B_{i,t'}^{\ell,u}(F_t)$ from $\mathfrak{B}$, the dual belief in the negated fact $B_{i,t'}^{\ell',u'}(\neg F_t)$ with $\ell' = 1 - u$ and $u' = 1 - \ell$ (cf. Corollary 3.4) has to be satisfied as well. For beliefs in rules $B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F, G))$, satisfiability depends on the accumulated subjective posterior interpretations of all threads weighted with their respective frequencies. The goal of the following procedure is to successively create threads for every belief in a fact $B_{i,t'}^{\ell,u}(F_t)$ in $\mathbf{T}_2(\mathfrak{B})$, such that we obtain representatives for the set of threads that (i) satisfy the respective fact F_t and for the set of threads that satisfy $\neg F_t$, and (ii) exhibit varying frequencies for all beliefs in temporal rules $B_{i,t'}^{\ell,u}(r_{\Delta t}^{\mathsf{fr}}(F, G)) \in \mathbf{T}_2(\mathfrak{B})$. Consequently, belief formulae can be considered as splitting rules and their application to generate representative threads results in a procedure similar to tableau-based methods. However, beliefs in temporal rules can induce splits both forward and backward in time and thus—unlike conventional tableau-based methods—the following procedure does not create a tree structure, but instead a set of sequences that represent possible threads. A key difference between the generation of representative threads and other logical sat solvers is that in PDT Logic it is virtually impossible to discard any generated potential thread: the probabilistic nature of the semantics requires that not only threads are considered where a given formula holds, but also threads where it does not. Thus, even threads violating the objects of given belief formulae are usually required to show satisfiability of a corresponding set of belief formulae $\mathfrak{B}$. The following discussion provides a general outline for a decision procedure in PDT Logic if only a set of belief formulae $\mathfrak{B}$ is given. An actual implementation of these methods is possible, but to obtain feasible run times for practical problems, various optimization techniques from research on logic reasoning implementations would need to be implemented, which is beyond the scope of this work.

4.5.1. Generating Representative Threads

Since the existence of any non-strict belief in a fact $B_{i,t'}^{\ell,u}(F_t)$ requires the existence of at least two threads—one, where the respective belief object is satisfied and one, where it is not[4]—we start with creating two threads from $\langle\hat{\Omega}_{\mathfrak{B}}(1), ..., \hat{\Omega}_{\mathfrak{B}}(t_{max})\rangle$ such that we obtain a set $\hat{\mathcal{T}} = \langle Th_1, Th_2\rangle$ with $Th_1 \models \varphi$ and $Th_2 \models \neg\varphi$ for all belief objects $\varphi = F_t$ contained in the set $\mathbf{T}_2(\mathfrak{B})$ to obtain a minimal set of set threads $\hat{\mathcal{T}}$ such that all belief formulae $\mathscr{B} \in \mathbf{T}_2(\mathfrak{B})$ can potentially be satisfied. This set will then subsequently be expanded with additional threads until either a suitable set of threads to show satisfiability of $\mathbf{T}_2(\mathfrak{B})$ is found, or until no more additional threads can be created.

To allow for a concise notation, in the following we adapt the frequency notation for all belief objects and use $(1 \cdot \varphi)$ to denote that φ is true, $(0 \cdot \varphi)$ to denote that φ is false, and generally $(x \cdot \varphi)$ to denote that φ holds with frequency x. Of course, values $0 < x < 1$ can only occur for belief objects that represent temporal rules. With this notation, we try to create initial sound threads such that

$$Th_1 \models \bigwedge_j (1 \cdot \varphi_j), \text{ and} \tag{4.30}$$

$$Th_2 \models \bigwedge_j (0 \cdot \varphi_j) \tag{4.31}$$

holds for the respective belief objects φ_j of all belief formulae $\mathscr{B}_j \in \mathbf{T}_2(\mathfrak{B})$.[5]

[4]Technically, a non-strict belief $B_{i,t'}^{\ell,u}(\varphi)$ could be satisfied with a single thread Th such that $Th \models \varphi$ if the belief's quantification has an upper bound $u = 1$. This might give rise to further optimizations for an actual implementation, but for the sake of simplicity, we do not consider this case explicitly.

[5]This notation is slightly simplified: for disjunctive belief formulae $\mathscr{B}_j = B_{i,t'}^{\ell,u}(\varphi_j') \vee B_{i,t'}^{\ell,u}(\varphi_j'')$, we use φ_j as an abbreviation for $\varphi_j' \vee \varphi_j''$.

This initial set $\hat{\mathcal{T}} = \{Th_1, Th_2\}$ is meant to represent the two extreme choices for possible threads with respect to $\mathbf{T}_2(\mathfrak{B})$ to provide a suitable starting point for the subsequently employed search heuristic. In general, it is not necessarily possible to create such extreme threads in compliance with (4.30) and (4.31) for every possible set of belief formulae $\mathbf{T}_2(\mathfrak{B})$. For instance, $\mathbf{T}_2(\mathfrak{B})$ might contain conflicting beliefs in facts $B^{\ell,u}_{i,t'}(F_t)$ and $B^{\ell,u}_{i,t'}(\neg F_t)$. Obviously, no single thread can satisfy both belief objects simultaneously, but it might still be possible to create a set of threads such that—together with a suitable probability assignment—both beliefs can be satisfied. Thus, (4.30) and (4.31) characterize the intended goal when creating the initial threads Th_1, Th_2, but do not represent hard constraints on these threads.

To find suitable threads that match these constraints, we employ the principle of least effort by adding as few facts as possible to each thread: For every belief in a fact $B^{\ell,u}_{i,t'}(F_t)$, we add the explicit constraints $F \in Th_1(t)$ and $F \notin Th_2(t)$, such that Th_1 represents the thread where all belief objects are true and Th_2 represents the set where all belief objects are false. For beliefs in rules $B^{\ell,u}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(F,G))$ we add $G \in Th_1(t+\Delta t)$ (resp. $\neg F \in Th_1(t-\Delta t)$) whenever another constraint enforces $F \in Th_1(t)$ (resp. $\neg G \in Th_2(t)$). If no occurrence of F respectively $\neg G$ is enforced in Th_1, a rule $r^{\mathsf{fr}}_{\Delta t}(F,G)$ is trivially satisfied with frequency 1 (i.e., there are no occurrences where F is not followed by G in Δt steps) and no further constraints need to be added. Analogously, for Th_2 we need to ensure that F holds at least once and that whenever $F \in Th_2(t)$ holds, $\neg G \in Th_2(t+\Delta t)$ holds, as well. For disjunctive belief formulae $B^{\ell,u}_{i,t'}(\varphi_1) \vee B^{\ell,u}_{i,t'}(\varphi_2)$, we need to ensure that belief object φ_1 or φ_2 holds in thread Th_1, as described above, and that $\neg\varphi_1 \wedge \neg\varphi_2$ holds in thread Th_2. If possible, the respective belief object φ_1 or φ_2 for thread Th_1 should be chosen such that no additional beliefs are triggered (we say that a belief is triggered by a fact F, if the existence of F enforces another constraint through a belief in a temporal rule or a disjunctive belief formula). Nested belief formulae are treated as above with respect to their innermost belief object. If some constraint cannot be applied because it is in conflict with previously added constraints from $\mathbf{T}_2(\mathfrak{B})$, it is simply skipped in this stage. As the creation of Th_1 and Th_2 is only the initialization step for a heuristic search of possible set threads, skipped constraints will still be considered later in subsequent expansions.

Whenever a constraint regarding a fact F is added to Th_1 or Th_2, it is necessary to check whether this triggers additional rules from set of type 1 beliefs $\mathbf{T}_1(\mathfrak{B})$. If necessary, resulting facts are added to the respective threads. This application works analogously to the construction of the set $\mathbf{T}_1^F(\mathfrak{B})$ as described in Section 4.4.1. Finally, if all belief formulae have been processed, we search for a sound thread with respect to the created constraints. Usually, a sound thread can be found easily by choosing all facts that are yet unconstrained in Th_1 and Th_2 such that they do not trigger any additional beliefs. Especially, for possible worlds $Th(t)$ that are unconstrained, we can choose $Th(t) = \emptyset$ if $\mathfrak{B}$ does not contain any belief in rules with purely negative preconditions or disjunctive belief formulae that are not satisfiable by $\emptyset$. More generally, the principle of least effort should be employed such that worlds ω are selected so that no further belief formulae need to be considered. Such a selection is impossible if and only if the addition of both F and $\neg F$ to some world triggers additional beliefs. Then, the consequences of adding the respective fact need to be evaluated, as well. The resulting set $\hat{\mathcal{T}} = \{Th_1, Th_2\}$ then provides a minimal set of representative threads that that can be used to check $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}})$.

In the following, we show how the principle of least effort can be used to obtain representative threads as efficiently as possible. The constraints used in the following example provide the minimal number of constraints that need to be enforced to obtain representative threads for the desired threads Th_1 and Th_2. For all worlds ω without any specific constraints, we simply use $\omega = \emptyset$. One can easily verify that this indeed yields threads in compliance with (4.30) and (4.31).

Example 4.8 (Trains continued)**.** We continue the train example with the sets of typed belief formulae specified in Example 4.5. In Example 4.7, it was shown that the set of worlds at time 1 $\hat{\Omega}_{\mathfrak{B}}(1)$ is restricted such that $\{at(T_1, C_A), on(A, T_1)\} \subseteq \omega$ for every world $\omega \in \hat{\Omega}_{\mathfrak{B}}(1)$. The set $\mathbf{T}_2(\mathfrak{B})$ contains three non-strict belief formulae, namely

$$\begin{aligned}
\mathbf{T}_2(\mathfrak{B}) = \{ & B_{A,0}^{.81,.81}\big(r_0^{pfr}(at(T_1, C_A), punct(T_1))\big), && (\mathscr{B}_2')\\
& B_{A,0}^{.81,.81}\big(r_0^{pfr}(at(T_2, C_C), punct(T_2))\big), && (\mathscr{B}_2'')\\
& B_{A,0}^{.93,.93}\big(r_2^{pfr}(Obs_{\{A\}}(\neg punct(train)), Obs_{\{AB\}}(\neg punct(train)))\big)\} && (\mathscr{B}_6)
\end{aligned}$$

By evaluating these belief formulae, we obtain constraints on the possible worlds in threads Th_1 and Th_2. A visualization of the following steps is given in Figure 4.2.

Analysis of belief formula $\mathscr{B}_2'$ results in the constraints $punct(T_1) \in Th_1(1)$ and $punct(T_1) \notin Th_2(1)$. These facts in turn trigger rules $\mathscr{B}_3'$ and $\mathscr{B}_3''$, respectively:

$$\begin{aligned}
& B_{A,0}^{1,1}\big(r_3^{pfr}(\ punct(T_1) \wedge at(T_1, C_A),\ at(T_2, C_C) \wedge on(A, T_2))\big) \text{ and} && (\mathscr{B}_3')\\
& B_{A,0}^{1,1}(r_5^{pfr}(\neg punct(T_1) \wedge at(T_1, C_A),\ at(T_2, C_C) \wedge on(A, T_2))\big), && (\mathscr{B}_3'')
\end{aligned}$$

resulting in the additional constraints $\{at(T_2, C_C), on(A, T_2)\} \subseteq Th_1(4)$ and $\{at(T_2, C_C), on(A, T_2)\} \subseteq Th_2(6)$.

Application of belief formula $\mathscr{B}_2''$ then yields the additional facts $\mathit{punct}(T_2) \in Th_1(4)$ and $\mathit{punct}(T_2) \notin Th_2(6)$. Again, this triggers rules from $\mathbf{T}_1(\mathfrak{B})$:

$$B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_2(\ \mathit{punct}(T_2) \wedge \mathit{at}(T_2, C_C),\ \mathit{at}(T_2, C_B) \wedge \mathit{on}(A, T_2))\big) \text{ and} \qquad (\mathscr{B}_4')$$
$$B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_3(\neg\mathit{punct}(T_2) \wedge \mathit{at}(T_2, C_C),\ \mathit{at}(T_2, C_B) \wedge \mathit{on}(A, T_2))\big), \qquad (\mathscr{B}_4'')$$

resulting in the additional constraints $Th_1(6) = \mathit{at}(T_2, C_B), \mathit{on}(A, T_2)$ and $Th_2(9) = \mathit{at}(T_2, C_B), \mathit{on}(A, T_2)$.

Note that belief formula

$$B^{1,1}_{A,0}\big(r^{\mathit{pfr}}_0(\neg\mathit{punct}(\mathit{train}) \wedge \mathit{at}(\mathit{train}, \mathit{city}), \mathit{Obs}_{\{A\}}(\neg\mathit{punct}(\mathit{train})))\big) \quad (\mathscr{B}_5)$$

from $\mathbf{T}_0(\mathfrak{B})$ provides a global constraint on the set of possible worlds $\hat{\Omega}_{\mathfrak{B}}$ such that $\mathit{Obs}_{\{A\}}(\neg\mathit{punct}(\mathit{train}))$ holds in every world where $\neg\mathit{punct}(\mathit{train})$ holds, and thus we obtain for thread Th_2 the additional facts $\mathit{Obs}_{\{A\}}(\neg\mathit{punct}(T_1)) \in Th_2(1)$ and $\mathit{Obs}_{\{A\}}(\neg\mathit{punct}(T_1)) \in Th_2(6)$.

Finally, rule

$$B^{.93,.93}_{A,0}\big(r^{\mathit{pfr}}_2(\mathit{Obs}_{\{A\}}(\neg\mathit{punct}(\mathit{train})), \mathit{Obs}_{\{AB\}}(\neg\mathit{punct}(\mathit{train})))\big) \qquad (\mathscr{B}_6)$$

does not change the created threads Th_1, Th_2: in Th_1 the rule's precondition is never enforced to be satisfied and thus the resulting frequency is one, while the lack of any observation in Th_2—even though there are nonpunctual trains—ensures that the resulting frequency is zero.

When trying to solve the resulting problem $\mathit{sat}(\mathbf{T}_2(\mathfrak{B}), \{Th_1, Th_2\})$, the non-strict belief formulae yield the following constraints on Th_1:

$$\begin{aligned} \mathscr{B}_2' :&\quad 0.81 \leq \mathcal{I}(Th_1) \leq 0.81 \\ \mathscr{B}_2'' :&\quad 0.81 \leq \mathcal{I}(Th_1) \leq 0.81 \\ \mathscr{B}_6 :&\quad 0.93 \leq \mathcal{I}(Th_1) \leq 0.93 \end{aligned}$$

Clearly, these constraints cannot be satisfied simultaneously and therefore the set $\hat{\mathcal{T}} = \{Th_1, Th_2\}$ is insufficient to show satisfiability of $\mathbf{T}_2(\mathfrak{B})$ (and therefore $\mathfrak{B}$).

If the created set of threads fails to show satisfiability of $\mathbf{T}_2(\mathfrak{B})$, additional threads

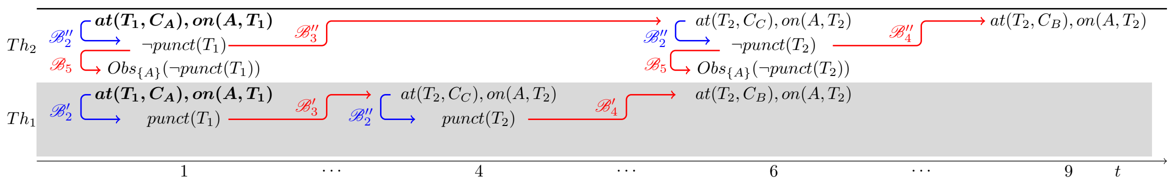


Figure 4.2.: Visualization of the representative thread set generation for the train example. Both threads start with the given facts $at(T_1, C_A), on(A, T_1)$. Applications of formulae from $\mathbf{T}_2(\mathfrak{B})$—such that Th_1 contains positive belief objects and Th_2 contains negative belief objects—are marked in blue, additional constraints from $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$ are marked in red.

can be created to continue searching for an expanded set $\hat{\mathcal{T}}$ such that $\mathbf{T}_2(\mathfrak{B})$ can be satisfied with respect to $\hat{\mathcal{T}}$. Based on an existing thread Th, an additional thread Th' can be created by ensuring that one conjunct for Th_1 or Th_2 in (4.30) and (4.31) is not satisfied anymore, i.e., from a given thread Th with existing constraints $(x_k \cdot \varphi_k)$, a new thread Th' can be obtained through the substitution

$$Th \models \bigwedge_j (x_j \cdot \varphi_j) \quad \Rightarrow \quad Th' \models \bigwedge_{j \neq k} (x_j \cdot \varphi_j) \ \wedge \ x'_k \cdot \varphi_k, \ x'_k \neq x_k. \tag{4.32}$$

Every such substitution of one conjunct with a new constraint provides a choice point to direct the continuation of the search for a suitable set of threads. The constraint notation in 4.32 is used to provide a formal characterization of choice points. In practice, a new thread Th' satisfying the above constraint can usually be created easily through the addition of new or the modification of existing facts in Th as follows. To simplify the following discussion, we assume that the expansion keeps a history of expansion steps and resulting consequences, such that all effects of adding an additional F can be undone if the respective fact F is changed for a newly created thread.

Definition 4.7 (Principle of least effort (ple) expansion)**.** Let $\hat{\mathcal{T}}$ be a set of threads and let $\mathbf{T}_2(\mathfrak{B})$ be a set of type 2 belief formulae. A *principle of least effort expansion* creates an expanded set $\hat{\mathcal{T}}' = \hat{\mathcal{T}} \cup \{Th'\}$ according to a single application of one of the following rules.

- For a (possibly negated) belief in an a fact $B^{\ell,u}_{i,t'}(F_t) \in \mathbf{T}_2(\mathfrak{B})$: If there exists a thread $Th \in \hat{\mathcal{T}}$ such that $F \in Th(t)$ (resp. $F \notin Th(t)$) is *not* yet enforced, Th' is created as a duplication of Th with the additional constraint $F \in Th(t)$ (resp. $F \notin Th(t)$).
- For a belief in a temporal rule $B^{\ell,u}_{i,t'}(r^{\mathsf{fr}}_{\Delta t}(F, G)) \in \mathbf{T}_2(\mathfrak{B})$: If there exists a thread $Th \in \hat{\mathcal{T}}$ such that $F \in Th(t)$ but $G \in Th(t + \Delta t)$ (resp. $G \notin Th(t + \Delta t)$) is *not* yet enforced, Th' is created as a duplication of Th with the additional constraint $G \in Th(t + \Delta t)$ (resp. $G \notin Th(t + \Delta t)$).

- For a disjunctive belief formula $\mathscr{B} = (B^{\ell,u}_{i,t'}(\varphi_1) \vee B^{\ell,u}_{i,t'}(\varphi_2) \vee \cdots) \in \mathbf{T}_2(\mathfrak{B})$: If possible, expansion is carried out with respect to one belief $B^{\ell,u}_{i,t'}(\varphi)$ as described in the two previous steps.
- Nested beliefs are again treated with respect to their innermost belief object.
- If the new thread Th' is created from Th through the addition of $F \in Th'$ for some fact F and time point t and $F \notin Th$ was enforced in the original thread Th, the consequences of adding $F \notin Th$ are undone in the new thread Th'.

  Then, for the created thread Th', additional belief formulae from $\mathbf{T}_1(\mathfrak{B})$ that are triggered by this modification need to be evaluated to obtain a sound thread, as described above for the creation of initial threads Th_1, Th_2.

The intuition behind this ple-expansion is to create additional threads that satisfy an alternative set of belief objects φ contained in the set $\mathbf{T}_2(\mathfrak{B})$ with as little effort as possible. In general, it is possible to add constraints on arbitrary facts at arbitrary time points and then continue with a successive expansion based on this thread. However, this would result in a rather aimless exploration of the exponential search space. Following the ple-expansion instead helps to direct the search for a suitable model guided by the rules specified in $\mathbf{T}_2(\mathfrak{B})$. To illustrate this, consider Figure 4.2 from the previous example: Possible ple-expansions could for example result in an additional thread by altering the punctuality of train T_2. Clearly, the resulting situations are intended in this model, as they were already considered in the original thread specification (cf. Figure 3.1). On the other hand, by deviating from the ple-expansion, one could add additional facts—say $at(T_1, C_A), on(A, T_1)$ at arbitrary time points $t > 1$. This could then give rise to multiple subsequent expansions of the resulting thread and may actually serve to generate a model for $\mathfrak{B}$, while such a situation was not intended by the specification of $\mathfrak{B}$. The example about train punctuality also illustrates the requirement of an undo operation: The fact $punct(T_2) \in Th_1(4)$ produced the additional constraint $\{at(T_2, C_B), on(A, T_2)\}$ at time $t = 6$. Clearly, this constraint should not be enforced any longer if—based on Th_1—a new thread Th' is created such that $punct(T_2) \notin Th'(4)$.

With information about violated constrains from the linear program corresponding to $sat(\mathbf{T}_2(\mathfrak{B}), \{Th_1, Th_2\})$, we can perform a dependency-directed selection of choice points: If the lower bound of a belief $B^{\ell,u}_{i,t'}(\varphi_k)$ cannot be satisfied with the current set of threads, an additional thread Th' can be created with the existing constraints on Th_1 or Th_2 and substituting the respective constraint on φ_k, as shown in (4.32).

The dependency of choice points on violated lower bounds can best be illustrated through the results from the previous example: Clearly, the upper bounds induced by $\mathscr{B}'_2$ and $\mathscr{B}''_2$ and the lower bound induced by $\mathscr{B}_6$ hinder satisfiability of $\mathbf{T}_2(\mathfrak{B})$ with respect to the created threads. Using the belief object of formula $\mathscr{B}'_2$ (or $\mathscr{B}''_2$) to create an additional thread Th_3 yields the updated constraint

$$\mathscr{B}_2': \qquad 0.81 \leq \mathcal{I}(Th_1) + x \cdot \mathcal{I}(Th_3) \leq 0.81$$

with a factor x depending on the frequency of the respective belief object in Th_3, while the constraint induced by $\mathscr{B}_6$ remains unchanged. As a result, the new constraint only allows for lower values of $\mathcal{I}(Th_1)$, and thus the lower bound induced by $\mathscr{B}_6$ remains unsatisfiable. Using the belief object of formula $\mathscr{B}_6$ to create an additional thread instead yields the constraint

$$\mathscr{B}_6: \qquad 0.93 \leq \mathcal{I}(Th_1) + x \cdot \mathcal{I}(Th_3) \leq 0.93,$$

which—through nonzero values for x and $\mathcal{I}(Th_3)$—potentially allows for lower values on $\mathcal{I}(Th_1)$. Note that this example only uses atomic belief formulae. For disjunctive belief formulae $\mathscr{B} = (B_{i,t'}^{\ell,u}(\varphi_1) \vee B_{i,t'}^{\ell,u}(\varphi_2) \vee \cdots)$, any of the respective belief objects with a violated lower bound can be used to direct the selection of subsequent choice points (given that no other disjunct of $\mathscr{B}$ is satisfiable, of course).

Combining information about violated lower bounds with the principle of least effort provides a multi-stage heuristic to proceed with a dependency-directed selection of choice points:

Definition 4.8 (Dependency-directed search heuristic)**.** Let $\mathbf{T}_2(\mathfrak{B})$ be a set of type 2 belief formulae and let $\hat{\mathcal{T}}$ be a set of threads such that $\neg sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}})$ holds. Then, to enable a dependency-directed search for an expanded set $\hat{\mathcal{T}}' \supset \hat{\mathcal{T}}$ such that $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}')$ holds, $\hat{\mathcal{T}}$ is expanded with an additional thread $Th' \notin \hat{\mathcal{T}}$ according to the following rules.

1. If the existing set of threads $\hat{\mathcal{T}}$ fails to satisfy lower bounds of constraints induced by a belief formula $\mathscr{B}$ with belief object φ and an additional thread Th' can be obtained through *one* ple-expansion with respect to φ, $\hat{\mathcal{T}}$ is expanded to $\hat{\mathcal{T}}' = \hat{\mathcal{T}} \cup \{Th'\}$.
2. Otherwise, if no dependency-directed ple-expansion is possible, another ple-expansion is applied to $\hat{\mathcal{T}}$, if possible.
3. Finally, if no ple-expansion is possible in $\hat{\mathcal{T}}$, an additional thread Th' can be created by adding the constraint $F \in Th(t)$ (resp. $F \notin Th(t)$) for arbitrary facts F that are not yet constrained in $Th(t)$.

The intuition behind this heuristic is that information about violated probabilistic constraints should be used to select a suitable next expansion step, if possible. Otherwise, other possible ple-expansion steps should be performed to use rules from $\mathbf{T}_2(\mathfrak{B})$

to guide the search. Only if no further ple-expansions are possible, additional constraints should be employed to continue the search. Restricting possible expansions with respect to criterion 1 to one step follows the principle of least effort, again: To illustrate this, consider Example 4.8: It was shown that the created set of threads $\{Th_1, Th_2\}$ fails to satisfy the lower bound of belief formula $\mathscr{B}_6$. In thread Th_1, there is no world $Th_1(t) \models Obs_{\{A\}}(\neg punct(train))$ such that the precondition of the rule in $\mathscr{B}_6$ is satisfied. Consequently, there is no single step ple-expansion of Th_1 that could change the constraints induced by $\mathscr{B}_6$. On the other hand, Th_2 provides two such choice points and should therefore be preferred for expansion. Note that the soundness requirement will determine choices for all unconstrained facts. Thus, in general the proposed expansion may produce threads that are already contained in $\hat{\mathcal{T}}$ by constraining facts that have been determined before. We will not consider this scenario explicitly but instead assume that in such cases, further expansion steps are performed until an additional thread is created.

4.5.2. A Thread Generation Example

To illustrate the expansion of a set of threads $\hat{\mathcal{T}}$ with respect to the dependency-directed search heuristic from Definition 4.8, in the following we resume the train example.

Example 4.9 (Trains continued)**.** In the previous example, a set of threads $\hat{\mathcal{T}} = \{Th_1, Th_2\}$ has been created that fails to show satisfiability of $\mathbf{T}_2(\mathfrak{B})$. Consequently, the heuristic from Definition 4.8 should be used to iteratively expand this set until an expanded set of threads $\hat{\mathcal{T}}'$ is created such that a model for $\mathfrak{B}$ is obtained or no further expansions of $\hat{\mathcal{T}}'$ are possible. Belief formula

$$\mathscr{B}_6 = B^{.93,.93}_{A,0}\big(r^{pfr}_2(Obs_{\{A\}}(\neg punct(train)), Obs_{\{AB\}}(\neg punct(train)))\big)$$

has already been identified as a belief formula which yields constrains with an unsatisfiable lower bound and this should therefore be used to guide the subsequent expansion. As already discussed before, no single-step ple-expansion of Th_1 is possible to influence the constraints induced by $\mathscr{B}_6$. Therefore we continue with an expansion based on thread Th_2. A visualization of the following steps is given in Figure 4.3.

There are two worlds in Th_2 where $Obs_{\{A\}}(\neg punct(train))$ is satisfied, namely $Obs_{\{A\}}(\neg punct(T_1)) \in Th_2(1)$ and $Obs_{\{A\}}(\neg punct(T_2)) \in Th_2(6)$. Both of these occurrences allow for an ple-expansion. We choose $Th_2(1)$ to perform the expansion. This yields a new thread Th_3 with the additional constraint $Obs_{\{A,B\}}(\neg punct(T_1)) \in Th_3(3)$, while all constraints from Th_2 remain intact, since there are no constraints that need to be undone by adding $Obs_{\{A,B\}} \in Th_3(3)$.

The expanded set $\hat{\mathcal{T}}' = \hat{\mathcal{T}} \cup \{Th_3\}$ can then be used to check $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}')$. In thread Th_3, the rule contained in $\mathscr{B}_6$ is satisfied in one of two occurrences of $Obs_{\{A\}}(\neg punct(train))$ and therefore yields a frequency of 0.5. Consequently, through transformation into a linear program we obtain the constraints

$$\begin{array}{llll}
\mathscr{B}_2' : & 0.81 \leq & \mathcal{I}(Th_1) & \leq 0.81 \\
\mathscr{B}_2'' : & 0.81 \leq & \mathcal{I}(Th_1) & \leq 0.81 \\
\mathscr{B}_6 : & 0.93 \leq & \mathcal{I}(Th_1) + 0.5 \cdot \mathcal{I}(Th_3) & \leq 0.93
\end{array}$$

Apparently, $\mathscr{B}_6$ allows for lower values of $\mathcal{I}(Th_1)$ for the this set $\hat{\mathcal{T}}'$. From the constraints induced by $\mathscr{B}_2'$ (resp. $\mathscr{B}_2''$) we still obtain $\mathcal{I}(Th_1) = 0.81$. Then, the constraint induced by $\mathscr{B}_6$ requires $\mathcal{I}(Th_3) = 0.24$ (since $0.81 + 0.5 \cdot 0.24 = 0.93$). This is still no suitable probability assignment since the sum over all priors exceeds one. Consequently, the thread set expansion continues. The above constraints show that—according to condition 1 of the search heuristic—thread Th_3 is now a suitable candidate for further expansion with respect to the belief object of $\mathscr{B}_6$.

Thus, based on Th_3, we create an additional thread Th_4 through ple-expansion. In this case, the only possible expansion step is $Obs_{\{A,B\}} \in Th_4(8)$, which results in a frequency of one for the rule contained in $\mathscr{B}_6$. Thus, testing $sat(\mathbf{T}_k(\mathfrak{B}), \hat{\mathcal{T}}')$ with the further expanded set $\hat{\mathcal{T}}'$ yields the following constraints:

$$\begin{array}{llll}
\mathscr{B}_2' : & 0.81 \leq & \mathcal{I}(Th_1) & \leq 0.81 \\
\mathscr{B}_2'' : & 0.81 \leq & \mathcal{I}(Th_1) & \leq 0.81 \\
\mathscr{B}_6 : & 0.93 \leq & \mathcal{I}(Th_1) + 0.5 \cdot \mathcal{I}(Th_3) + 1 \cdot \mathcal{I}(Th_4) & \leq 0.93
\end{array}$$

These constraints are now satisfiable, for instance with

$$\mathcal{I}(\hat{\mathcal{T}}') = \big(0.81,\ 0.07,\ 0,\ 0.12\big).$$

Thus, $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}')$ returns a positive result and satisfiability checking of $\mathfrak{B}$ can terminate with this result.

This result concludes satisfiability testing of the set of belief formulae $\mathfrak{B}$ originally specified in Example 4.1. Nevertheless, for illustration purposes we show the result of further applications of ple-expansion steps in Figure 4.4. Changes in the additionally created threads are obtained through a further respectively different application of a

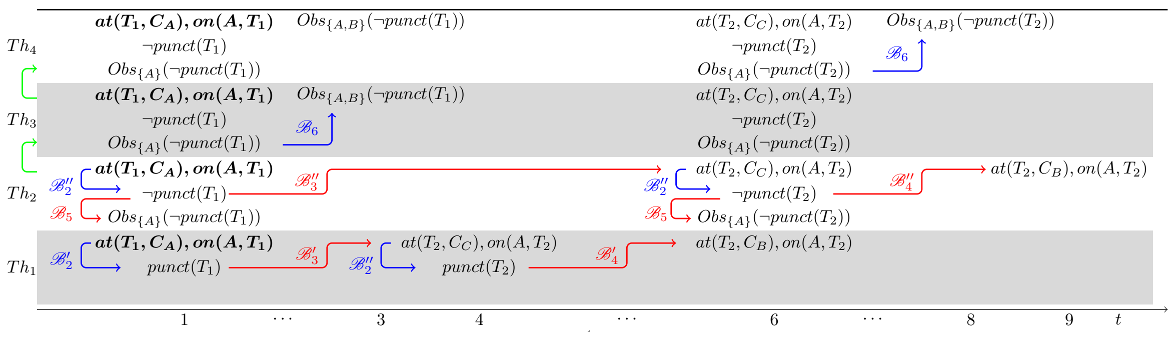

Th_4
$\boldsymbol{at(T_1, C_A), on(A, T_1)}$
$\neg punct(T_1)$
$Obs_{\{A\}}(\neg punct(T_1))$
$Obs_{\{A,B\}}(\neg punct(T_1))$
$at(T_2, C_C), on(A, T_2)$
$\neg punct(T_2)$
$Obs_{\{A\}}(\neg punct(T_2))$
$Obs_{\{A,B\}}(\neg punct(T_2))$
$\mathscr{B}_6$
Th_3
$\boldsymbol{at(T_1, C_A), on(A, T_1)}$
$\neg punct(T_1)$
$Obs_{\{A\}}(\neg punct(T_1))$
$Obs_{\{A,B\}}(\neg punct(T_1))$
$\mathscr{B}_6$
$at(T_2, C_C), on(A, T_2)$
$\neg punct(T_2)$
$Obs_{\{A\}}(\neg punct(T_2))$
Th_2
$\mathscr{B}''_2$
$\mathscr{B}_5$
$\boldsymbol{at(T_1, C_A), on(A, T_1)}$
$\neg punct(T_1)$
$Obs_{\{A\}}(\neg punct(T_1))$
$\mathscr{B}''_3$
$\mathscr{B}''_2$
$\mathscr{B}_5$
$at(T_2, C_C), on(A, T_2)$
$\neg punct(T_2)$
$Obs_{\{A\}}(\neg punct(T_2))$
$\mathscr{B}''_4$
$at(T_2, C_B), on(A, T_2)$
Th_1
$\mathscr{B}'_2$
$\boldsymbol{at(T_1, C_A), on(A, T_1)}$
$punct(T_1)$
$\mathscr{B}'_3$
$\mathscr{B}''_2$
$at(T_2, C_C), on(A, T_2)$
$punct(T_2)$
$\mathscr{B}'_4$
$at(T_2, C_B), on(A, T_2)$
1
…
3
4
…
6
…
8
9
t

Figure 4.3.: Visualization of ple-expansions for the train example. Applications of formulae from $\mathbf{T}_2(\mathfrak{B})$ are marked in blue, additional constraints from $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$ are marked in red. Expansion steps are marked in green.

belief formula from $\mathbf{T}_2(\mathfrak{B})$, marked in blue in the respective threads. Worlds $Th(t)$ that remain unconstrained after a saturated application of ple-expansions are marked with "/". All of these worlds then give rise to further expansions according to step 3 of the search heuristic.

Some comments on the resulting set of threads from this example are necessary. Comparing the final threads depicted in Figure 4.4 with the original set of threads introduced in Figure 3.1 shows that the expansion result largely corresponds to the original specification (except differing thread labels). There are some notable differences however.

- First of all, there is the additional predicate *punct*(*train*), which was introduced in Example 4.1 to allow for a concise specification of the background knowledge. As the concept of nonpunctual trains (and especially the respective ramifications) are implicitly encoded in Figure 3.1 as well, this does not change the properties of the modeled example.

- With the explicit representation of train punctuality, observations of nonpunctual trains can be expressed explicitly in this example, while the previous example uses the ramifications of nonpunctual trains to model observations. Since rules $\mathscr{B}_3$ and $\mathscr{B}_4$ assert that ramifications of punctual respectively nonpunctual trains are common knowledge among Alice and Bob, both modeling alternatives preserve the intended meaning of the example.

- Another difference is the timing of Alice's observations. In the original example we assumed that such an observation occurs at the time point when a train was supposed to arrive at the destination city. In the current example we assume that Alice already observes that a train is not punctual when leaving the departure city. The reason for this change is solely for illustration purposes: specifying in rule $\mathscr{B}_5$ that Alice immediately observes a nonpunctual train yields a type 0 belief

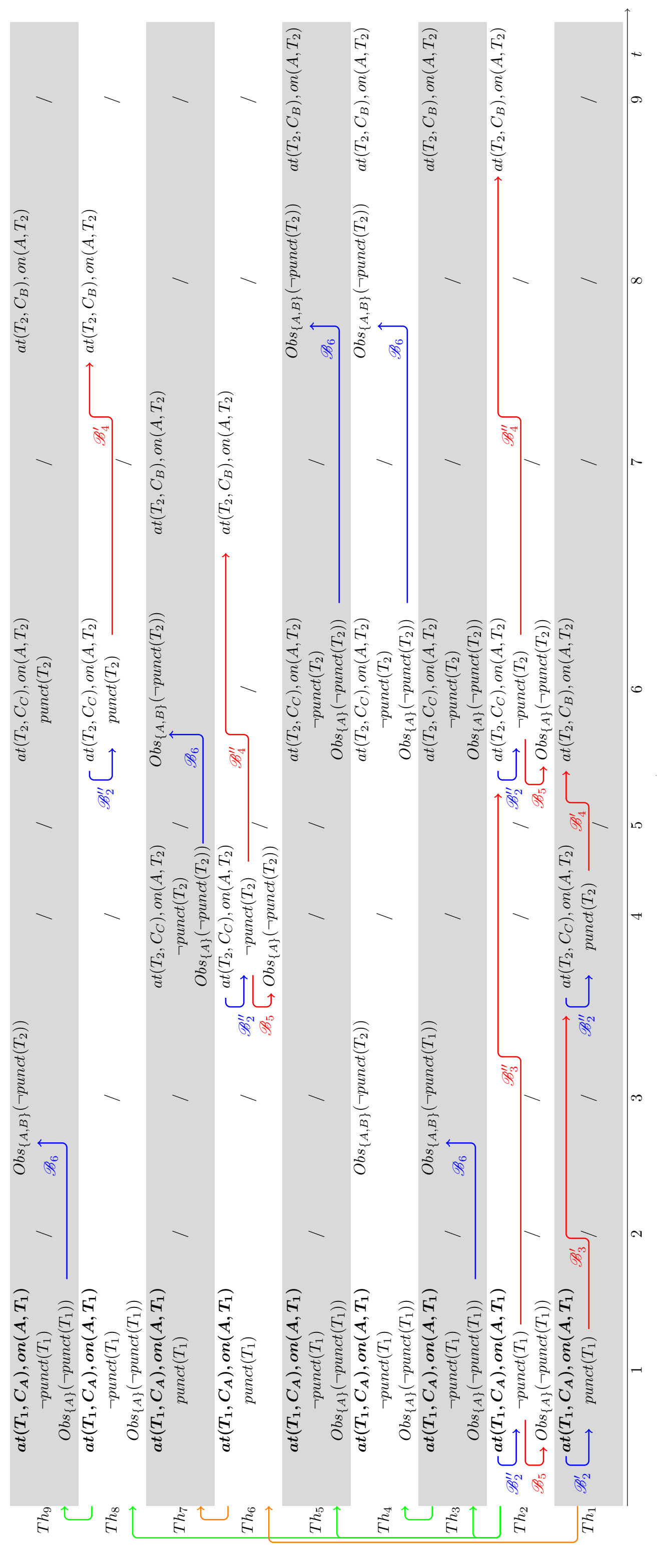


Figure 4.4.: Visualization of continued ple-expansions for the train example. Applications of formulae from $\mathbf{T}_2(\mathfrak{B})$ are marked in blue, additional constraints from $\mathbf{T}_0(\mathfrak{B})$ and $\mathbf{T}_1(\mathfrak{B})$ are marked in red. Expansion steps originating from Th_1 and Th_2 are marked in green and orange, respectively. Unconstrained worlds are marked with “/”.

and thus serves to illustrate how additional facts can be obtained through global constraints. Since rule $\mathscr{B}_6$ ensures that potential calls to Bob (i.e., shared observations) occur two time points after Alice's original observation, the intended model of the original example is still maintained.

- The above points are only concerned with specific details of the modeled domain. Comparing the set threads from Figure 3.1 with the threads from Figure 4.4 also shows a more general modeling problem: for instance, analyzing the worlds at time point 2 in Figure 4.4 shows that Alice is *not* (necessarily) on train T_1, while she is on this train both at the previous time point and later time points. Naturally, one should expect that Alice is on the train at all intermediate time points between boarding and exiting the train. This is an instance of the *frame problem* (e.g., [Rei01]) that occurs when specifying dynamic systems through logic formulae. Generally, the frame problem is concerned with finding a suitable set of axioms to describe adequate evolutions of the world. From a modeling perspective, evolutions where Alice vanishes and reappears while on a train ride are obviously no adequate evolutions of the world. An application of the final step of the search heuristic could then yield a tremendous blow-up of the considered set of threads. For the modeled problem, this would clearly result in *unintended models*, but the resulting models could still serve to show satisfiability of the respective set of belief formulae $\mathfrak{B}$, even though this result might not be desired. This problem could be fixed by adding successor state axioms in the style of [Rei01], e.g., specifying that if Alice is on a train, she remains there for the next time point unless she explicitly exits the train.

4.5.3. Properties of the Representative Thread Generation

In this section, we provide results to connect the set of representative threads to the satisfiability problem of PDT Logic and discuss the complexity of generating representative threads.

Theorem 4.15 (PDT Logic Decision Procedure). *Let $\mathfrak{B}$ be a set of PDT Logic belief formulae, and let $\hat{\mathcal{T}} = \{Th_1, Th_2\}$ be the initial set of threads with length t_{max} obtained from $\mathfrak{B}$ according to Equations (4.30) and (4.31). Iteratively expanding this set according to the search heuristic from Definition 4.8 and testing $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}')$ for the expanded sets $\hat{\mathcal{T}}'$ until (i) $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}')$ returns a positive result, or (ii) $\hat{\mathcal{T}}'$ is fully expanded with respect to the search heuristic yields a sound and complete decision procedure for $sat(\mathfrak{B}, t_{max})$.*

Proof. Both the initial set of threads $\hat{\mathcal{T}} = \{Th_1, Th_2\}$ and the expanded sets $\hat{\mathcal{T}}'$ obtained through ple-expansion steps are defined such that only sound threads according to Definition 4.4 are considered. Theorem 4.13 states that the decision problem $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}})$ is equivalent to $sat(\mathfrak{B}, \hat{\mathcal{T}})$ if the set $\hat{\mathcal{T}}$ contains only threads that are sound with respect to $\mathfrak{B}$. A positive result for $sat(\mathfrak{B}, \hat{\mathcal{T}})$ for threads with length t_{max} shows that $\hat{\mathcal{T}}$ is a model for $\mathfrak{B}$ and thus $sat(\mathfrak{B}, t_{max})$ follows. Consequently, a positive result for $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}')$ always proofs that $\mathfrak{B}$ is satisfiable for t_{max} time points.

On the other hand, if no model for $\mathfrak{B}$ has been found and it is not possible to create additional threads according to the search heuristic from Definition 4.8, the search space is fully explored. From this it follows that no model for $\mathfrak{B}$ with t_{max} time points exists and therefore $\mathfrak{B}$ is unsatisfiable for t_{max} time points. Consequently, it follows that the PDT Logic decision procedure is sound.

With these properties, the completeness result is straightforward: For any arbitrary input $\mathfrak{B}$ and t_{max}, either a model can be found or non-existence of such a model can be proven through a full exploration of the search space, and thus completeness of the procedure follows. □

In the following, we analyze the complexity of generating representative threads for a set of belief formulae $\mathfrak{B}$.

Theorem 4.16 (Complexity of representative thread generation). *Let $\mathfrak{B}$ be a set of belief formulae. Creating a set of expanded representative threads $\hat{\mathcal{T}}'$ for $\mathfrak{B}$ is in EXPSPACE.*

Proof. The maximum number of possible threads for a given set of belief formulae $\mathfrak{B}$ is determined through the size $|\mathbf{F}_{\mathfrak{B}}|$ and the maximum time point t_{max}. Recall from Equation (4.3) that we use $\mathbf{F}_{\mathfrak{B}}$ to identify all event formulae from $\mathfrak{B}$ and use this as the set of ground atoms to construct possible worlds. Since every belief operator contains at most two event formulae, we obtain the constraint $|\mathbf{F}_{\mathfrak{B}}| \leq 2{\cdot}k{\cdot}\mathfrak{B}$ for k belief operators in $\mathfrak{B}$. The largest set of possible threads is then obtained as the sequences of combinations of all possible worlds over all time points, yielding $2^{2 \cdot k \cdot t_{max} \cdot \mathfrak{B}}$ possible threads. In the worst case, all $2^{2 \cdot k \cdot t_{max} \cdot \mathfrak{B}}$ representative threads are created before obtaining a satisfiability result. Consequently, creating all possible representative threads is in the complexity class *DSPACE*$(2^{p(n)})$, which is the class *EXPSPACE*. □

From this theorem, we immediately obtain complexity results for the satisfiability problem $sat(\mathfrak{B}, t_{max})$.

Corollary 4.17 (Complexity of PDT SAT without a given set of threads). *Checking satisfiability of a set of PDT Logic belief formulae $\mathfrak{B}$ without a specification of possible threads is in EXPSPACE.*

Proof. The generation of representative threads is in *EXPSPACE*, as shown in Theorem 4.16. For a given set of threads Theorem 4.11 shows that satisfiability checking in PDT Logic is in NP. Thus, this does not further increase complexity of the PDT sat problem without a given set of threads and it follows that this problem is in *EXPSPACE*. □

Some comments on these results are necessary. Since the decision procedure outlined in Theorem 4.15 yields an exponential expansion of possible threads $\hat{\mathcal{T}}'$—which all need to be fed into the decision problem $sat(\mathbf{T}_2(\mathfrak{B}), \hat{\mathcal{T}}')$—the exponential space requirement is evident. However, as we have illustrated with the example, positive satisfiability results can possibly be already obtained through small sets of possible threads $\hat{\mathcal{T}}'$ with a diminutive size compared to the entire search space. Moreover, the discussion of the train example has shown that a major part of the search space stems from insufficient rule specifications. This is not a specific problem of our formalism nor the presented decision procedure, but a general problem of rule-based modeling approaches, namely the aforementioned frame problem. An incomplete model specification then leads to the generation of *unintended models*, which serve to show satisfiability of the modeled problem, but have not been intended by the respective modeler. This could lead to the worst case—both from a complexity and from a model perspective—that after an exponential execution of the decision procedure, the result only shows that the input specification does not specify the intended model. The problem can be addressed on the modeling side by providing additional axioms to ensure that no unintended model is generated. However, this leads to a significant increase in the specification size and it is difficult to ensure through rule specifications that indeed every unintended model is prevented.

The ple-expansion steps could be used as a heuristic to discriminate between intended and unintended models: As shown in the train example, only applying ple-expansion steps results in a relatively small set of threads, which indeed corresponds to the intention of the model, while any further expansions inherently leads to an exponential growth of the set of threads and introduces only additional unintended models. Thus, omitting the final step of the search heuristic would give a significantly faster termination of the decision procedure, even though the resulting procedure cannot prove unsatisfiable sets of formulae any longer. However, one could use the expansion procedure to create the set of intended threads first and—possibly after an inspection by the modeler—continue to use this set to perform satisfiability checks with respect to the intended model.

The runtime of the expansion procedure and resulting satisfiability checks is clearly tilted towards the positive side: If a set of belief formulae is satisfiable, there is a good chance that satisfiability can be shown in a small number of steps. Negative results on the other hand can only be obtained after an exhaustive exploration of the search space. However, for many applications negative satisfiability results are required. For instance, checking entailment $\mathfrak{B} \models \mathscr{B}$ can be checked through the reformulation $\neg sat(\mathfrak{B} \cup \neg\mathscr{B})$. For applications relying on such a reformulation, the presented procedure is unfavorable because positive entailment results can never be obtained efficiently. One could overcome this problem as sketched above by generating a set of intended threads first and then use this set to perform subsequent satisfiability tests—once a set of threads is given, the decision problem's complexity significantly decreases, as shown in Section 4.3.

4.6. Concluding Remarks

This chapter discusses alternative ways of specifying problems in PDT Logic, either through explicit enumerations of possible threads with specific probabilities or through a set of appropriate rules. Both approaches exhibit their specific advantages and drawbacks: For many problem domains, relying on an exhaustive enumeration of all possible threads poses a severe obstacle for modeling the respective scenarios, as the specification is practically unmanageable. On the other hand, there are problem domains (e.g., attack graphs in cyber security scenarios, as discussed in the following chapter) that come with such an explicit specification anyways. If probability assignments are given as well, we have shown that it is possible to check given models very efficiently for these types of problems.

To overcome modeling disadvantages of the thread-based approach, we have also shown how a problem domain can be solely specified through a set of PDT Logic belief formulae. For most problem domains, this is a more natural way of specifying the problem. Also, this provides means to easily adapt many existing problems—that are specified in other formal languages as sets of rules—to PDT Logic. On the other hand, waiving the requirement of an exhaustive thread specification and according probabilities extremely increases the problem complexity of checking satisfiability of a set of PDT Logic formulae. Nevertheless, even when only imprecise probabilities are given, the resulting problem remains decidable and the increased complexity might be curtailed through search heuristics.

Combinations of both approaches are possible as well: If an exhaustive specification of possible threads is given, but probability intervals are only specified through beliefs with imprecise probabilities, the satisfiability problem can be transformed into a 0-1 mixed integer linear program. As there are a variety of efficient solvers available for this class of problems, this transformation provides a means to exploit existing optimizations to check satisfiability of PDT Logic formulae.

Chapter 5

Analyzing Attackers' Beliefs: An Application Example in Cyber Security Domains

To further illustrate how PDT Logic can be put to practical use, we return to the cyber security example that we introduced informally in the introduction. In this chapter, we briefly summarize required preliminaries from the cyber security domain and then show how PDT Logic can be applied to the analysis of threat scenarios. For the discussion of this example, we follow conventions from the security literature and use the terms *attacker* and *intruder* interchangeably. We refer to the attacker as *she* and to the defender as *he*.

5.1. Threat Analysis with Attack Graphs

Information infrastructures have been exposed to an increasing number of cyber attacks in recent years. Since such information infrastructures are used in a variety of critical application areas, there is an increasing need to defend them against any potential cyber attacks. It is important to note that, due to external constraints, established security measures—such as patching known vulnerabilities—are only applicable to a limited extent. Legal requirements might allow only the use of certified software versions, or new patches cannot be applied at a certain point in time because compatibility tests are still pending. A common effect is that external constraints leave critical information infrastructures exposed to known vulnerabilities, at least for some time.

An established approach to analyze threats from identified vulnerabilities is the use of attack graphs.[1] An attack graph is a formal representation of all possible attack sequences in an IT infrastructure system. Analyzing such attack graphs provides means

[1]We will use *attack graph* as a general term for formal sequence-based approaches of attack analyses. Depending on the specific properties of the respective formalization, these approaches are known under a variety of names such as attack graphs, attack defense trees, attack countermeasure trees, or attack response trees.

to identify vulnerabilities of the system and to decide on suitable countermeasures. Several approaches of automatically creating attack graphs have been proposed, e.g., [PS98], [OGA05], and [ILP06]. Methods of analyzing threats in cyber security scenarios and identifying suitable countermeasures based on attack graphs have been discussed for instance in [NJ08], [RKT10], [RKT12], and [LIS$^+$06].

Standard reactions to such an analysis include the proactive removal of identified vulnerabilities. However, several scenarios exist where an application of these measures is not feasible without impairing the mission of the organization responsible for the critical infrastructure. Therefore, it is important to carefully analyze potential consequences of applicable countermeasures or sequences of countermeasures (as part of defense strategies). Maintaining a model of an intruder's belief state provides the defender with improved means to assess the merits of potential defense strategies.

Existing research on attack graph analysis relies on exhaustive specifications of possible attack sequences and several approaches to obtain such attack graphs automatically have been proposed. Thus, attack graphs provide a natural way to use PDT Logic for threat analysis. Extending existing attack graphs with a formal model of both the intruder's and defender's belief states allows for analyzing pending threats on an additional level. Since these approaches originate from exhaustive specifications of all possible attack sequences (i.e., threads in PDT Logic), we can apply the computationally cheap model checking algorithm (Algorithm 1 from Chapter 4) to analyze the evolution of belief states in this domain. To simplify the discussion of our example, we assume that the intruder has already breached the network and is carrying out attacks on specific systems. Thus, we only analyze the final stages of a known attack graph. Of course, in general we can apply PDT Logic to all stages of an attack graph, but this would significantly blow up the set of threads that we need to consider. Since a short sequence of actions suffices to illustrate the application of PDT Logic in cyber security scenarios, we keep the example as simple as possible and concentrate on the final stages of an attack, where immediate consequences are pending.

5.2. Formalizing Agents' Beliefs

In the following, we discuss properties of our intended target domain, provide a formal model of this domain and show how the beliefs of both intruders and defenders evolve.

5.2.1. Considerations on the Target Domain

As discussed previously, we are concerned with situations where pre-emptive security measures are not always an option. Thus, the network might be exposed to known vulnerabilities and the defender is left with choosing the best reactive countermeasure in case of an attack. We start our analysis at the final stages of an attack graph and assume that any attacker breaching this point has already obtained extensive information about the network. To simplify the presentation of the example, we assume that at any time point only a single intruder is attacking the system.

For our example, we assume that we have a single attacker model and that any attack (e.g., a remote code execution) to the network consists of (at least) two actions: First, the attacker has to gain access to a target system with sufficient privileges. Then, custom code can be executed on this system to reach the attacker's actual goal. We do not model the details of these steps, but abstractly represent the first step as an *attack* on a system resulting in access to a shell (e.g., through exploitation of known vulnerabilities), and the second step as some *code execution* on the target system. After having successfully obtained a shell on the target system, the attacker basically has two options: either she can proceed with the second stage of her attack (i.e., code execution) or she can try to gain access to further systems. Both options come with advantages and drawbacks for the intruder: continuing to attack further systems might result in additional compromised systems, but at the same time decreases the chance of performing an attack undetected. The choice of action depends on the attacker's actual goal; she might even attack another system without actually executing code there, but only to create distractions from her actual goal.

A network based intrusion detection system (IDS) can be used to detect attack actions on specific systems. However, in practice no IDS is perfect, i.e., missed attacks have to be considered when employing an IDS. For the sake of simplicity, potential false alerts of the IDS are not considered in our example. Considering attacks that are not detected by an IDS is an important point when planning defense strategies: if every detected attack is countered with a corresponding defense action, the lack of such a defense lets the attacker *know* that her attack went undetected and she might proceed with executing malicious code without having to fear any actions from the defender. Furthermore, deliberately letting the attacker execute code on a non-critical target host can provide valuable insights: an analysis of the executed code will reveal the actual goal of the attack and might further reveal the identity of the attacker. Another reason for refraining from a defense operation is that this action (e.g., unplugging a control server) might impact the mission success just as much as an attack. Consequently, deliberately letting an observed attack pass undefended might provide higher expected utility for the defender. By analyzing the potential evolution of the attacker's belief states, the choice of not defending can even be used to drive the attacker to false conclusions regarding her success.

Continuing these considerations, it might prove useful for the defender to maintain some kind of "honey pot" within the network. In its classical form, a honey pot is a system that has no productive meaning but is used instead to attract attackers and thereby provides means to analyze their goals and identities. We adopt the concept of honey pots to our model by maintaining backup devices of critical systems. These backup devices are disconnected from the physical world but are otherwise indistinguishable from the actual productive system. This way, an intruder does not know which one the critical system is, but if she executes malicious code on the honey pot, she is not able to impact the mission success, but instead unknowingly provides the defender with the possibility to analyze the code and identify the attacker.

5.2.2. An Exemplary Domain Model

In the following, we introduce a small example to show how we can formally model potential attacks in a computer network and apply PDT Logic to analyze the evolution of the agents' belief states.

Our scenario contains two agents, the defender D and the intruder I. Following the considerations from the previous section, we assume that two systems are present in this network: some honey pot A and the corresponding critical system B. Possible actions on a system X are denoted by *attack(X)* and *defend(X)* with the obvious meanings. Furthermore, execution of malicious code on a system X is denoted by *exec(X)*. Finally, we have observation atoms such as $Obs_{\{D\}}(attack(A))$, indicating that the defender observed an attack on system A.

Building on these events, we can construct a set of threads representing all possible event sequences in this example. The resulting set is depicted in Figure 5.1.

Remark 5.1. In Figure 5.1, we have used a tree notation instead of a set of parallel threads. This is only for an easier depiction of the situation, and a transformation into the usual set of threads is straightforward: whenever two threads share a common prefix sequence, we have depicted this sequence as a single path in the tree. By explicitly representing the paths from the root node to every leaf node as a single thread, we obtain the usual thread notation.

This model represents our considerations from the previous section: analysis starts at some time when no attack has occurred yet ($t = 0$). Possible subsequent events are then attacks on system A or system B (represented through nodes 53 and 78 in the graph) or no attacks (node 1). If an attack has occurred on, say, system A, the IDS can detect this attack (i.e., an $Obs_{\{D\}}(attack(A))$ occurs, represented through the solid outgoing edges from node 53), or the attack is not detected (represented through the dashed line). For an undetected attack, the defender obviously has no options to defend against this. For an observed attack, the defender can choose between defending against this attack (node 55) or deliberately refrain from a defense (node 56). A defense (with potential

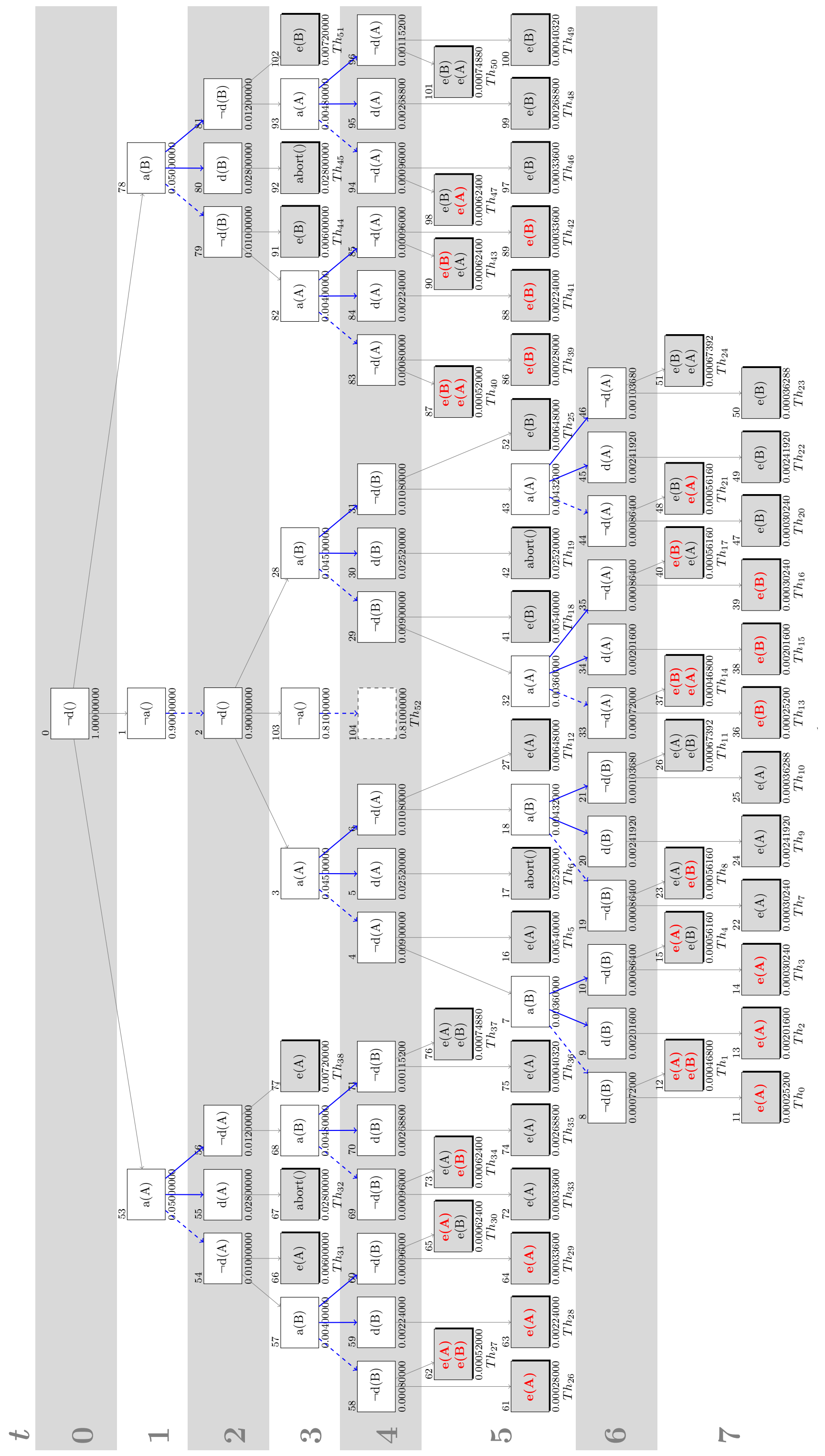


Figure 5.1.: Possible threads for the example domain. a(X), d(X), and e(X) denote attack, defend and code execution actions on node X, respectively. Observations of the defender are marked through solid blue edges, lack of an observation with a dashed blue edge. Unobserved code executions are marked in red, otherwise they are marked in black. Terminal nodes of threads are marked in gray.

downsides to the defender's mission) forces the intruder to abort his attack. Formally, this means that a defense action on system X always induces a shared observation $Obs_{\{A,B\}}(defend(X))$. These observations are not explicitly specified in Figure 5.1, but to express that a defense action always corresponds to a shared observation, we can simply specify the following rule as a type 0 belief (cf. the belief formula taxonomy from Section 4.4.1):

$$B^{1,1}_{A,0}(r^{pfr}_0(defend(X), Obs_{\{A,B\}}(defend(X)))) \tag{5.1}$$

Lack of a defense action gives the intruder two options again: she can execute her malicious code on the attacked system (nodes 66 and 77), or she can proceed to attack the other system (nodes 57 and 68). After a second attack, possible subsequent events match the ones discussed for the first attack. Finally, if the first attack has not been defended, there are various options for the intruder to execute malicious code: if the second attack is defended, the attacker can execute the code only on the previously attacked system (nodes 63 and and 74), otherwise she can choose between executing code only on the previously attack system (e.g., node 61) or on both systems (e.g., node 62). If an attack has been detected by the IDS, the defender is able to observe these code executions (denoted in black), for undetected attacks, subsequent code executions will remain undetected as well (marked in bold red in the graph). If no attacks occur at all, the system continues in its normal state as indicated through node 104—allowing for equivalent branchings of the graph if attacks occur at later points in time.

Note that in most states of this model, none of the agents has complete knowledge of the world. For instance, if the defender does not observe an attack at time $t = 2$, he cannot distinguish between undetected attacks on either system or the actual absence of an attack (i.e., he considers nodes 54, 2, and 79 as actually being possible). If the intruder actually attacked a system and does not observe a defense action, she does not know whether her attack was actually undetected or whether it was observed and intentionally not defend (i.e., she is unable to distinguish between nodes 54 and 56, or 79 and 81, respectively).

To assign probabilities to every thread in this scenario, we start with assigning probabilities to single events. Then, we can determine the probability of a thread as joint probabilities of the respective events contained in this thread. Reasonable values of single events—which could for example be obtained through an analysis of existing log data—are given in Table 5.1. The resulting probabilities of individual nodes are depicted in Figure 5.1.

Table 5.1.: Single event probabilities for the cyber security example

Event	Probability
Attack on a specific system at a specific time point	0.05
Attack detected by the IDS	0.80
Defend action after an observed attack	0.70
Second attack after a successful first attack	0.40
Code execution on both nodes after two attacks	0.65

5.2.3. An Analysis of Belief Evolutions

To illustrate how we can use this example to analyze the evolution of both agents' beliefs, let us assume that the intruder has attacked system A at time $t = 3$ (i.e., the actual world at $t = 3$ is represented through node 3). If the defender has observed this attack, he has to decide whether to defend the system against this attack or not. However, even if the defender observed this attack, he is not able to identify the actual node in the attack graph, because he is unable to distinguish between the situation where $attack(A)$ is the first action (node 3) and the situation where this was preceded by an undetected attack on B (node 82). To analyze the defender's expectations for the worst case (an unobserved code execution on the productive system B, i.e., $exec(B) \wedge \neg Obs_{\{D\}}(exec(B))$), one can analyze the threads containing nodes 5 or 84, and 6 or 85 in Figure 5.1, respectively. By summing over the normalized probabilities of these respective threads, where the terminal node contains $exec(B) \wedge \neg Obs_{\{D\}}(exec(B))$, one can verify that the following holds:

$$\begin{aligned} \neg defend(A) \text{ at } t = 4 &\models B_{D,4}^{0,\ 0.05}(exec(B) \wedge \neg Obs_{\{D\}}(exec(B))), \text{ and} \\ defend(A) \text{ at } t = 4 &\not\models B_{D,4}^{0,\ 0.05}(exec(B) \wedge \neg Obs_{\{D\}}(exec(B))), \end{aligned}$$

i.e., the defender has lower expectations in the worst case actually happening if he choses to not defend this attack. Consequently, we assume that the defender decides against a defend action, and we consider node 6 as the actual node for the following discussions. Next to protecting the system against the worst case, the defender is also interested in getting opportunities to analyze the intruder's malicious code. We can express his beliefs in observing a code execution (i.e., $Obs_{\{D\}}(exec(A)) \vee Obs_{\{D\}}(exec(B))$) as

$$\neg defend(A) \text{ at } t = 4 \quad \models B_{D,4}^{0.9,\ 1}(Obs_{\{D\}}(exec(A)) \vee Obs_{\{D\}}(exec(B))).$$

If the defender chooses not to take any defensive action, the intruder in turn is unable to distinguish between the situations where the defender deliberately took no action and where the defender simply missed the attack, i.e., the defender considers all threads possible that contain node 6 or node 4. Thus, the intruder has the following belief in actually being able to execute malicious code on a target system without being detected:

$$\neg Obs_{\{i\}}(\mathit{defend}(A)) \text{ at } t = 4 \quad \models \; B_{I,4}^{0,\,0.2}(\phi),$$

$$\text{with } \phi = \neg Obs_{\{D\}}(\mathit{exec}(A)) \wedge \neg Obs_{\{D\}}(\mathit{exec}(B)) \wedge (\mathit{exec}(A) \vee \mathit{exec}(B))$$

Since both the intruder and defender know the possible attack sequences, it is also possible to analyze belief states of the respective opponent: In the considered situation, the defender knows that the attacker has not observed any defense action and can therefore not distinguish between nodes 4 and 6. However, since the defender was not able to rule out a previous attack on B, from his point of view the intruder could still also consider nodes as 83, 85, 94, and 96 as possible (these are the nodes where B was attacked before and no defend action was taken). Still, the defender has a rather high belief in the intruder's actual belief state, as expressed in the following nested belief:

$$B_{D,4}^{0.8,\,1}(B_{I,4}^{0.0,\,0.2}(\phi))$$

This analysis shows that reasoning about nested beliefs provides the defender with quite an accurate estimation the attacker's actual belief state. This representation of nested beliefs can then aid the attacker in choosing the best countermeasures.

5.3. Concluding Remarks

This example from the cyber security domain illustrates how PDT Logic can be applied in practice to formalize multi-agent beliefs in a security context. This formal representation enables the analysis of the adversary's belief evolution depending on specific actions. Through a small example we have demonstrated how a formal belief state analysis can be carried out. Next to formal representations of both the intruder's and defender's beliefs, this is especially useful to gain an opportunity to reason about nested beliefs. This provides novel means of assessing the expected utility of any action when planning a defense strategy: Along with analyzing the direct effect of any action on the network, we can also analyze how any action will influence the belief state of the opponent. With the use of more sophisticated attack models, this enables the defender to drive the intruder into desired safe states (from the defender's point of view), where the intruder expects to achieve her goal, but is actually unable to cause real harm.

This chapter shows how PDT Logic can be used in practical scenarios by analyzing the effects of certain actions on the belief states of agents. So far, we have explicitly analyzed the outcomes of any potential action. To automatize the process of finding the best set of actions to achieve a desired goal automatically, we formalize abduction in PDT Logic in the following chapter.

Chapter 6

Abduction in PDT Logic

Abduction, or inference to the best explanation, is a form of reasoning where some data is given and a hypothesis is inferred that explains this data [Jos96]. Usually, the abduction problem is phrased as *given some background knowledge* $\mathfrak{B}$ *and an observation* $\mathscr{G}$*, abductive reasoning is the process of deriving a set of explanations* $\mathfrak{S}$ *of* $\mathscr{G}$ *according to* $\mathfrak{B}$. Formally, this can be expressed as

$$sat(\mathfrak{B} \cup \mathfrak{S}), \tag{6.1}$$
$$\mathfrak{B} \cup \mathfrak{S} \models \mathscr{G}, \tag{6.2}$$

where (6.1) states that the explanation $\mathfrak{S}$ has to be consistent with respect to the background knowledge, and (6.2) models that the hypothesis entails the observed data $\mathscr{G}$. $\mathfrak{S}$ is called a solution to the abduction problem if both (6.1) and (6.2) are satisfied.

Abduction has been a subject of extensive research (e.g., [EG95] and [JJ96]), with extensions to temporal logic (e.g., [Bar00]) and uncertainty (e.g., [Poo97]). However, there is little work that studies abduction in the context of both time and uncertainty. A recent study of abduction in APT Logic involving both time and uncertainty has been introduced in [MSS14]. This approach considers abduction for the single–agent case and uses time-invariant probabilities. By extending this work such that probabilistic multi-agent beliefs and their dynamic evolution can be represented, we develop a novel abductive formalism that is able to determine necessary actions to induce desired beliefs in a multi-agent scenario.

In the context of PDT Logic, we can reformulate the abduction problem as *given some background knowledge* $\mathfrak{B}$ *and a goal* $\mathscr{G}$*, what can we do to achieve* $\mathscr{G}$*?* This reformulation yields an alternative application to abductive reasoning: instead of finding a hypothesis to explain some observed data, we are interested in finding a hypothesis to bring about a certain goal state. Yet the formal problem expressed in (6.1) and (6.2) remains the same and thus these two formulations represent alternative views on the application of the abduction procedure, while the underlying formalism stays the same.

A formal representation of this problem in PDT Logic enables us to determine a required minimal set of actions that one has to take in order to bring about a desired goal belief state. The analysis of the cyber security example in Chapter 5 already showed how this type of reasoning can be useful in practice without having a formal account of the abduction problem yet. Next to cyber security settings, this approach may be useful in various other domains. To name only a few examples, in financial markets it might be critical for a company to determine what kind of information has to be released to the public such that the shareholders' belief in a positive outlook is sufficiently high. In cooperative multi–agent scenarios, it is useful to determine minimal required communication acts among agents, such that all agents obtain all relevant information. Generally speaking, abduction in PDT Logic enables a formal treatment of many information management problems.

Since PDT Logic is a framework to reason about probabilistic beliefs in multi-agent systems, a natural goal for abduction in PDT Logic is to bring about a specific belief state of some agent. As the semantics of the belief operator is defined with respect to the subjective posterior interpretations of the respective agent, it is clear that beliefs change according to the interpretation updates as given in Definition 3.9. The interpretations in turn are updated with the occurrence of observations and thus it is clear that the beliefs of an agent can be influenced by ensuring that the respective agent makes certain observations. We will use this below to identify possible actions to induce the abduction goal.

Before formalizing the abduction problem in PDT Logic, we need a few preliminary definitions. Since the following abduction procedure performs multiple satisfiability checks with respect to varying sets of formulae that all use the same set of possible threads, we assume that background knowledge is specified through a set of belief formulae $\mathfrak{B}$ together with a set of possible threads $\hat{\mathcal{T}}$. Such a set of threads might for example be created by generating an intended model heuristically, as discussed in Chapter 4. We use $\mathbf{K} = \langle \mathfrak{B}, \hat{\mathcal{T}} \rangle$ to denote this background knowledge. With a slight abuse of notation, we use $\mathbf{K} \cup \mathfrak{B}'$ as an abbreviation for $\langle \mathfrak{B} \cup \mathfrak{B}', \hat{\mathcal{T}} \rangle$ to denote that a set of belief formulae $\mathfrak{B}'$ is added to the set $\mathfrak{B}$ specified as background knowledge. Furthermore, we use $sat(\mathbf{K})$ as an abbreviation for $sat(\mathfrak{B}, \hat{\mathcal{T}})$, i.e., for satisfiability of the set of belief formulae $\mathfrak{B}$ with respect to the set of threads $\hat{\mathcal{T}}$, as discussed in Section 4.3.

With this notion, we can formulate the entailment relation between $\mathbf{K}$ and a belief formula $\mathscr{B}$ as a satisfiability problem in the usual way [Etc99], provided that $\mathbf{K} \cup \{\mathscr{B}\}$ is consistent:

Definition 6.1 (Entailment of belief formulae)**.** Let $\mathbf{K} = \langle \mathfrak{B}, \hat{\mathcal{T}} \rangle$ be the abduction background knowledge, and let $\mathscr{B}$ be a belief formula. Then, $\mathbf{K}$ entails $\mathscr{B}$ (denoted by $\mathbf{K} \models \mathscr{B}$) iff

$$\mathbf{K} \models \mathscr{B} \equiv \neg sat(\mathbf{K} \cup \{\neg \mathscr{B}\}).$$

With these preliminary definitions we are now able to provide a formal account of the abduction problem in PDT Logic.

6.1. Formalizing the Abduction Problem in PDT Logic

We have shown in the previous chapter that we can use a set of PDT Logic belief formulae $\mathfrak{B}$ to describe a specific scenario. Given such a scenario, it is often useful to know what actions one could take to induce a certain belief $B^{\ell,u}_{i,t'}(\varphi)$ of some agent at a specific time t'. From the semantics introduced in Chapter 3 it follows that beliefs of an agent can only change due to observations of the respective agent and hence it is natural to define possible actions as a set of observations that can be induced.

Definition 6.2 (PDT Logic Abduction)**.** Let $\mathbf{K}$ be some background knowledge, H be a set of PDT Logic formulae representing observations $Obs_{\mathcal{G}}(l)_t$ and let $\mathscr{G} = B^{\ell,u}_{i,t_g}(\varphi)$ be an atomic belief formula. Then, the triple $\langle \mathbf{K}, H, \mathscr{G} \rangle$ is an instance of the PDT Abduction Problem. $\mathfrak{S} \subseteq H$ is a solution to the abduction problem iff $\mathbf{K} \cup \mathfrak{S}$ is satisfiable and $\mathbf{K} \cup \mathfrak{S} \models \mathscr{G}$. A solution $\mathfrak{S}$ is a minimal solution to the abduction problem if there exists no solution $\mathfrak{S}'$ with $|\mathfrak{S}'| < |\mathfrak{S}|$ so that $\mathfrak{B} \cup \mathfrak{S}' \models \mathscr{G}$.

Intuitively, $\mathbf{K}$ constitutes the background knowledge that models a specific environment, $\mathscr{G}$ describes the goal we want to achieve, and the hypothesis space H represents information that we can share with the agents in order to induce the belief described by $\mathscr{G}$. Viewing the hypothesis space H as a set of possible observations for an agent simplifies the following considerations. However, in many scenarios an agent does not control observations directly, but instead choses to carry out specific actions which in turn induce certain observations. For instance, consider the cyber security example from Chapter 5 again: whenever the defender choses to defend an observed attack, a common observation $Obs_{\{A,B\}}(defend(X))$ of this action occurs. Thus, the defender does not chose to induce the respective observation explicitly, but instead this observation is a necessary consequence of his action (cf. rule (5.1), which states that every defend action inevitably yields a corresponding observation). Such connections between chosen actions and corresponding observations are domain-specific and cannot be treated in a general way. Still, the respective observations are the responsible elements that influence the goal belief state $\mathscr{G}$. Thus, rules such as (5.1) can be used to map domain-specific actions to the corresponding required observations of the abduction procedure.

Remark 6.1. Definitions 6.1 and 6.2 assume that a problem specification in PDT Logic is given through a set of threads $\hat{\mathcal{T}}$ and a set of belief formulae $\mathfrak{B}$ yielding constraints on the prior probabilities. As discussed in Chapter 4, instead of specifying $\mathfrak{B}$, we can alternatively specify a specific prior distribution. Since suitable decision procedures for both approaches are provided in Chapter 4 (cf. Theorems 4.1 and 4.10), these approaches both serve as a suitable problem specification for PDT Logic. To enable a concise presentation, we only discuss results with respect to a given set of belief formulae $\mathfrak{B}$ as input. We use a running example based on specific probability assignments to illustrate that this approach works as well, but we do not discuss the obvious adaptions for this input formally.

To illustrate a potential application of abduction in PDT Logic, consider the following example from stock markets, which is inspired by a related example given in [SPSS11] (without any treatment of beliefs however).

Example 6.1 (stock markets)**.** Assume a company C which is listed on the stock exchange. The set of all market participants is treated as a single agent P. In stock-related news feeds there is a rumor *rumor* at, say, time point 1 that due to some allegations, a government investigation into C's business conduct is pending. It is likely that—without any response from the company—such a rumor will let market participants consider this company as not trustworthy anymore, which will probably lead to a significant decrease *decr* of the company's stock price in the near future of, say, 5 time units, as modeled by the following rule (recall from Section 4.4.1 that prior beliefs are commonly known and can therefore be bound to an arbitrary agent):

$$B_{P,0}^{0.6,1}(r_0^{pfr}(rumor,\ decr)) \tag{6.3}$$

Naturally, company C has the desire to prevent this decrease if possible and has various options to react upon this rumor. The most basic reaction is to stay passive and simply see what will happen. In this case, only the above rule governs the expectation in a significant decrease of the company's stock price.

A second option would be to contact the government (modeled by *contactG*) directly to try to resolve the issues without involving the public. Of course, this results in an observation $Obs_{\{C\}}(contactG)$, but as this process does not happen in public, market participants are unaware of this. Consequently, they do not observe anything and the above rule is still the only one to govern the expectations of the company's stock price. The corresponding observation model is described through

$$B_{P,0}^{1,1}(r_0^{pfr}(contactG,\ Obs_{\{C\}}(contactG))) \tag{6.4}$$

To influence public opinion, the company essentially has two options: it can issue a press release (*pr*) explaining the situation, or the company's CEO can take the blame for the allegations and resign (*ceoRes*). Of course, these options can be combined to achieve different results. The following rules describe expected outcomes of these actions. If needed, we present the following rules in pairs: the first rule describes observation model of the respective action, and the second rule describes the consequences of the corresponding observation, respectively. To keep the example simple, we assume that any responsive action is taken directly after the rumor occurs (i.e., at time point 2) and that it corresponds to the same time point for the decrease as above (i.e., now 4 time units later).

First, the company can issue a press release *pr* explaining its own view on the allegations without taking any further actions. Any explanation has the potential to prevent the stock price decrease. However, without any additional measures, the impact of a press release is limited in comparison to the initial situation modeled in (6.3), as expressed by the following rule:

$$B^{1,1}_{P,0}(r^{pfr}_0(pr,\ Obs_{\{C,P\}}(pr))) \tag{6.5}$$

$$B^{0.5,0.8}_{P,0}(r^{pfr}_4(Obs_{\{C,P\}}(pr),\ decr)) \tag{6.6}$$

Instead of issuing a press release, the CEO can decide to simply resign without any comment. This would be considered by market participants as a clear admission of guilt, which in turn makes it impossible that a stock price decrease can be prevented:

$$B^{1,1}_{P,0}(r^{pfr}_0(ceoRes,\ Obs_{\{C,P\}}(ceoRes))) \tag{6.7}$$

$$B^{1,1}_{P,0}(r^{pfr}_4(Obs_{\{C,P\}}(ceoRes) \wedge \neg Obs_{\{C,P\}}(pr),\ decr)) \tag{6.8}$$

Next, the CEO can decide to resign but issue an according press release explaining current events. In this case, market participants might consider the resignation as an act of taking responsibility and thus as a first step in restoring trust in the company. Thus, we might obtain—for example by analyzing historic data from related situations—that the probability of a stock price decrease can be reduced a little further compared to the situation expressed in (6.6):

$$B^{0.3,0.6}_{P,0}(r^{pfr}_4(Obs_{\{C,P\}}(ceoRes) \wedge Obs_{\{C,P\}}(pr),\ decr)) \tag{6.9}$$

Corresponding observation models for these actions are already specified in (6.5) and (6.7).

If the company decides to contact the government to resolve the allegations, it might be useful to issue a corresponding press release in order to let market participants know that they are actively working on solutions to the current problem.

This will significantly reduce the probability of a pending stock price decrease. Of course, if the company issues a corresponding press release, there will be a shared observation between market participants P and company C. This results in the following rule with corresponding observation model:

$$B^{1,1}_{P,0}(r^{pfr}_0(pr \wedge contactG,\ Obs_{\{C,P\}}(pr) \wedge Obs_{\{C,P\}}(contactG))) \qquad (6.10)$$

$$B^{0,0.4}_{P,0}(r^{pfr}_4(Obs_{\{C,P\}}(pr) \wedge Obs_{\{C,P\}}(contactG),\ decr)) \qquad (6.11)$$

This concludes our specification of the stock market domain. Note that we have not specified probabilities for every possible combination of shared observations—the remaining combinations are not of interest for our example. The respective probabilities can be chosen arbitrarily as long as they do not violate some of the rules given above. To give a clear overview of the assumptions presented in this example, we summarize the relations between shared observations of specific facts and the resulting probability intervals for an expected price decrease in Table 6.1.

A possible set of threads $\hat{\mathcal{T}}$ for this example together with possible resulting Kripke structures for market participants P is depicted in Figure 6.1. To verify that this set of threads indeed satisfies all of the above rules, Table 6.2 provides all precise beliefs that can occur for market participants in this set of threads, depending on the respective Kripke structure. At time $t = 1$, P considers every thread possible (i.e., there is one Kripke structure $\mathcal{K}_{P,1}$ containing the entire set $\hat{\mathcal{T}}$) and thus, the according probability is obtained by simply summing over the probabilities of all threads Th_i where $Th_i(5) \models decr$ holds. For the resulting Kripke structures at time $t = 2$, the update rule (cf. Definition 3.9) has to be applied in order to obtain the respective beliefs. We will return to these beliefs later when discussing the abduction procedure in detail.

Now, a natural goal for company C is to convince market participants P that there is nothing to worry about, i.e., C aims at making P believe with a rather high degree of certainty that the stock price will not significantly decrease in the near future. A suitable goal can be modeled as

$$\mathscr{G} = B^{0.7,1}_{P,2}((\neg decr)_5) \qquad (6.12)$$

6.2. The Hypothesis Space

As the background knowledge $\mathbf{K}$ contains a set of possible threads $\hat{\mathcal{T}}$, we do not need to specify the hypothesis space H explicitly, but instead we can determine a set of

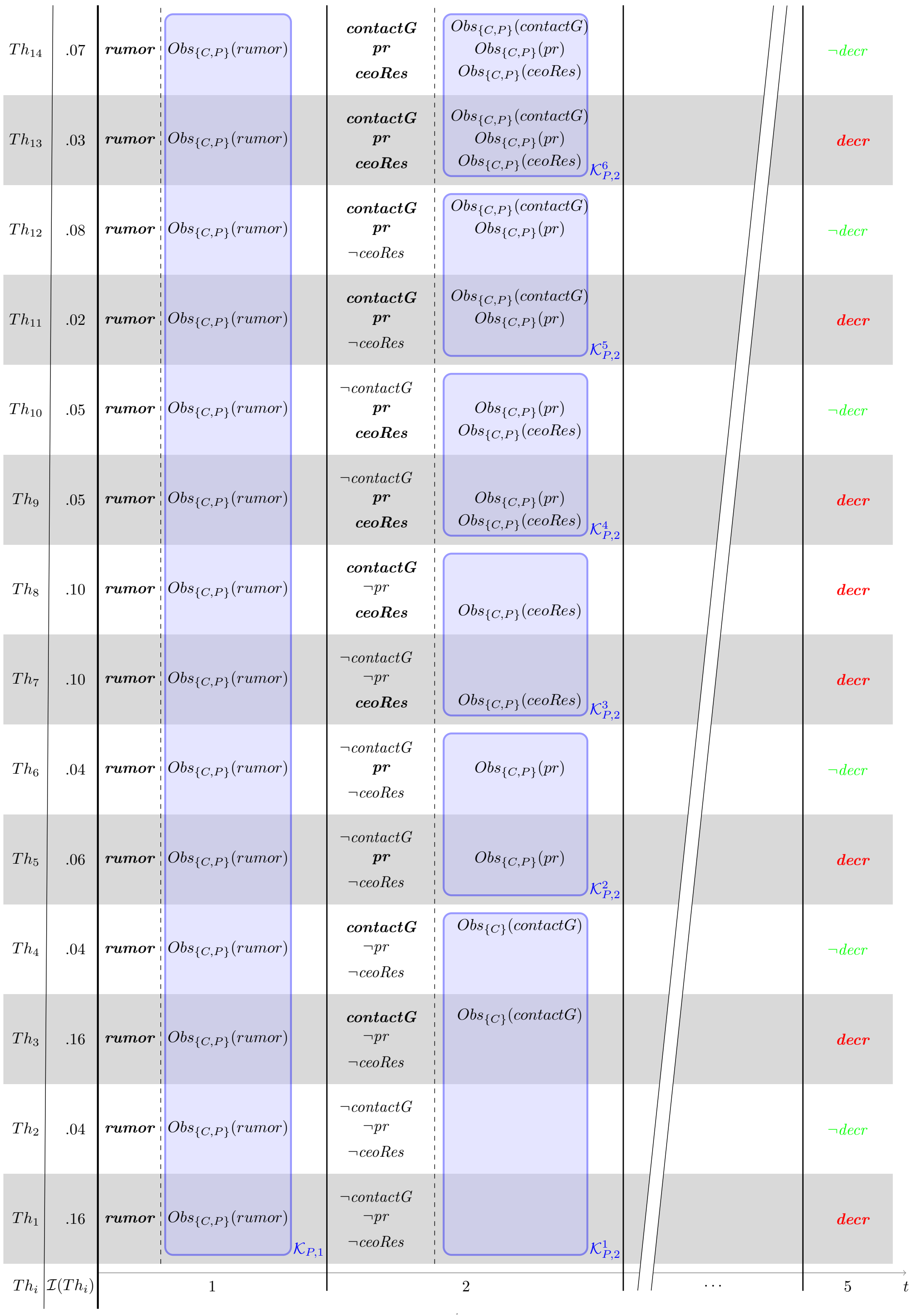

Th_{14} .07 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathbf{contactG}$ $\mathbf{pr}$ $\mathbf{ceoRes}$ $Obs_{\{C,P\}}(contactG)$ $Obs_{\{C,P\}}(pr)$ $Obs_{\{C,P\}}(ceoRes)$ $\neg decr$
Th_{13} .03 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathbf{contactG}$ $\mathbf{pr}$ $\mathbf{ceoRes}$ $Obs_{\{C,P\}}(contactG)$ $Obs_{\{C,P\}}(pr)$ $Obs_{\{C,P\}}(ceoRes)$ $\mathcal{K}^6_{P,2}$ $\mathbf{decr}$
Th_{12} .08 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathbf{contactG}$ $\mathbf{pr}$ $\neg ceoRes$ $Obs_{\{C,P\}}(contactG)$ $Obs_{\{C,P\}}(pr)$ $\neg decr$
Th_{11} .02 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathbf{contactG}$ $\mathbf{pr}$ $\neg ceoRes$ $Obs_{\{C,P\}}(contactG)$ $Obs_{\{C,P\}}(pr)$ $\mathcal{K}^5_{P,2}$ $\mathbf{decr}$
Th_{10} .05 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\neg contactG$ $\mathbf{pr}$ $\mathbf{ceoRes}$ $Obs_{\{C,P\}}(pr)$ $Obs_{\{C,P\}}(ceoRes)$ $\neg decr$
Th_9 .05 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\neg contactG$ $\mathbf{pr}$ $\mathbf{ceoRes}$ $Obs_{\{C,P\}}(pr)$ $Obs_{\{C,P\}}(ceoRes)$ $\mathcal{K}^4_{P,2}$ $\mathbf{decr}$
Th_8 .10 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathbf{contactG}$ $\neg pr$ $\mathbf{ceoRes}$ $Obs_{\{C,P\}}(ceoRes)$ $\mathbf{decr}$
Th_7 .10 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\neg contactG$ $\neg pr$ $\mathbf{ceoRes}$ $Obs_{\{C,P\}}(ceoRes)$ $\mathcal{K}^3_{P,2}$ $\mathbf{decr}$
Th_6 .04 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\neg contactG$ $\mathbf{pr}$ $\neg ceoRes$ $Obs_{\{C,P\}}(pr)$ $\neg decr$
Th_5 .06 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\neg contactG$ $\mathbf{pr}$ $\neg ceoRes$ $Obs_{\{C,P\}}(pr)$ $\mathcal{K}^2_{P,2}$ $\mathbf{decr}$
Th_4 .04 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathbf{contactG}$ $\neg pr$ $\neg ceoRes$ $Obs_{\{C\}}(contactG)$ $\neg decr$
Th_3 .16 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathbf{contactG}$ $\neg pr$ $\neg ceoRes$ $Obs_{\{C\}}(contactG)$ $\mathbf{decr}$
Th_2 .04 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\neg contactG$ $\neg pr$ $\neg ceoRes$ $\neg decr$
Th_1 .16 $\mathbf{rumor}$ $Obs_{\{C,P\}}(rumor)$ $\mathcal{K}_{P,1}$ $\neg contactG$ $\neg pr$ $\neg ceoRes$ $\mathcal{K}^1_{P,2}$ $\mathbf{decr}$
Th_i $\mathcal{I}(Th_i)$ 1 2 ... 5 t

Figure 6.1.: A possible set of threads for Example 6.1: at time 1 the rumor about company C occurs, and at time point 2 the company has several options to react. The company's possible actions and the resulting observations are depicted. Moreover, the resulting Kripke structures $\mathcal{K}^j_{P,t}$ for market participants P at time points t are depicted in blue. For every thread Th_i, the respective prior probability $\mathcal{I}(Th_i)$ is given and the outcome at time 5 is marked.

Table 6.1.: Single event probabilities for the example

Shared Observations $Obs_{\{C,P\}}(\cdot)$	**Probability Interval**
$\emptyset$	$[0.7, 1]$
pr	$[0.5, 0.8]$
$ceoRes$	$[1, 1]$
$pr,\ ceoRes,$	$[0.3, 0.6]$
$contactG,\ pr$	$[0, 0.4]$

Table 6.2.: Probabilistic beliefs of market participants in a price decrease. The objects of shared observations that induce each of the Kripke structures are given in parentheses.

	Kripke structure		***P*'s belief in *decr***
$t = 1:$	$\mathcal{K}_{P,1}$	$(rumor)$	0.68
$t = 2:$	$\mathcal{K}^1_{P,2}$	$(\emptyset)$	0.8
	$\mathcal{K}^2_{P,2}$	(pr)	0.6
	$\mathcal{K}^3_{P,2}$	$(ceoRes)$	1.0
	$\mathcal{K}^4_{P,2}$	$(pr, ceoRes)$	0.5
	$\mathcal{K}^5_{P,2}$	$(contactG, pr)$	0.2
	$\mathcal{K}^6_{P,2}$	$(contactG, pr, ceoRes)$	0.3

hypothesis candidates H' from $\hat{\mathcal{T}}$ as the set of all observations that can possibly occur:

$$H' = \{Obs_{\mathcal{G}}(l)_t : (\exists Th \in \hat{\mathcal{T}} : Obs_{\mathcal{G}}(l) \in Th(t))\} \tag{6.13}$$

Example 6.2 (stock markets continued)**.** Proceeding with the example described above, we can identify the set of hypothesis candidates from Figure 6.1 as

$$H' = \left\{ \begin{array}{l} Obs_{\{C\}}(contactG), \\ Obs_{\{C,P\}}(contactG), \\ Obs_{\{C,P\}}(pr), \\ Obs_{\{C,P\}}(ceoRes) \end{array} \right\}. \tag{6.14}$$

Before actually trying to solve an instance of the abduction problem specified in Definition 6.2, we can identify necessary preconditions that an observation $Obs_{\mathcal{G}}(l)_t \in H'$ has to satisfy in order to be able to contribute to a solution of the abduction problem: The set H' collects all observations that can possibly occur in the situation described by $\mathbf{K}$. However, not all of these observations have the means to alter the quantification of the goal belief $\mathscr{G}$. With a slight abuse of notation, we use $i \in \mathscr{G}$ to denote that agent i is involved in the goal belief $\mathscr{G}$, i.e., $\mathscr{G}$ contains a belief operator $B_{i,t'}^{\ell,u}$ (possibly as part of a nested belief). Then, we can define a dependency property $dep(\mathscr{G}, Obs_{\mathcal{G}}(l)_t)$ between the goal and an observation as follows:

Definition 6.3 (Goal dependency)**.** Let $\mathscr{G}$ be the abduction goal and let $Obs_{\mathcal{G}}(l)_t$ be an observation. $\mathscr{G}$ is dependent on $Obs_{\mathcal{G}}(l)_t$, denoted by $dep(\mathscr{G}, Obs_{\mathcal{G}}(l)_t)$, iff

$$i \in \mathscr{G} \ \wedge \ i \in \mathcal{G} \tag{6.15}$$

Naturally, any observation $Obs_{\mathcal{G}}(l)_t \in H'$ that does not satisfy this dependency property is unable to contribute to achieving the goal and can therefore be neglected when searching for a solution to the abduction problem. Thus, we can define the set of relevant atomic hypotheses as

$$H = \{Obs_{\mathcal{G}}(l)_t \in H' : \ dep(\mathscr{G}, Obs_{\mathcal{G}}(l)_t)\} \tag{6.16}$$

Example 6.3 (stock markets continued)**.** The example's goal as specified in (6.12) is to induce a specific belief for market participants P. Analyzing the observations in H' from Example 6.2 shows that observation $Obs_{\{C\}}(contactG)$ is unable to influence this goal, as P is not involved in this observation. Consequently, the abduction goal is only dependent on the set of observations

$$H = \left\{ \begin{array}{l} Obs_{\{C,P\}}(contactG), \\ Obs_{\{C,P\}}(pr), \\ Obs_{\{C,P\}}(ceoRes) \end{array} \right\}. \tag{6.17}$$

Whenever an observation occurs for some agent i, the set of threads it considers possible is reduced such that only those threads remain where the respective observation holds. We use $K_i^S(t_g)$ to denote the set of all possibility relations that can occur in different threads for agent i at the time t_g of the goal belief[1] induced by a potential solution $\mathfrak{S} \subseteq H$. We can then leverage the semantics of the belief operator (cf. Definition 3.11) to obtain another necessary precondition: for $\mathscr{G} = B_{i,t_g}^{\ell,u}(\varphi)$ with $0 < \ell$ and $u < 1$, in every distinguishable situation (t_g) that i considers possible at time t_g, there need to be two threads Th_1, Th_2 so that the respective belief object φ is satisfied in one thread and unsatisfied in another. For a belief with $u = 1$, there

[1]To simplify the presentation, we assume that (even for nested beliefs) the goal formula $\mathscr{G}$ involves only a single time point t_g. The proposed methods are also applicable to goal formulae involving multiple time points, but this will significantly increase the complexity of presentation.

needs to be at least one thread such that the respective belief object φ is satisfied. If the belief is quantified with $\ell = u = 1$, all threads in all distinguishable situations (t_g) have to satisfy the belief object φ.[2] Otherwise, if these conditions are not met, it is clear that the goal belief is not entailed, independently of any specific probability assignment. These conditions can be checked prior to evaluating the semantic entailment $\mathbf{K} \cup \mathfrak{S} \models \mathscr{G}$. Using $poss(S)$ to denote the syntactic possibility of a solution $\mathfrak{S}$, we can formally express these considerations as

$$poss(\mathfrak{S}) = \begin{cases} & \text{for } 0 < \ell, u < 1 \text{ and } \forall (t_g) \in K_i^S(t_g): \\ true & \quad \text{if } \exists Th_1, Th_2 \in (t_g) : Th_1(t_g) \models \varphi \wedge Th_2(t_g) \models \neg\varphi \\ false & \quad \text{otherwise} \\ & \text{for } u = 1 \text{ and } \forall (t_g) \in K_i^S(t_g): \\ true & \quad \text{if } (\exists Th \in (t_g) : Th(t_g) \models \varphi) \\ false & \quad \text{otherwise} \\ & \text{for } \ell = u = 1 \text{ and } \forall (t_g) \in K_i^S(t_g): \\ true & \quad \text{if } (\forall Th \in (t_g) : Th(t_g) \models \varphi) \\ false & \quad \text{otherwise} \end{cases} \tag{6.18}$$

With these considerations we can define the entire search space $\mathfrak{H}$ for possible solutions to the abduction problem as

$$\mathfrak{H} = \{\mathfrak{S} \in 2^H : \; poss(\mathfrak{S})\} \tag{6.19}$$

Example 6.4 (stock markets continued)**.** In Figure 6.1, the subset of threads $\mathcal{K}_P^3$ induced by solution candidate $Obs_{\{C,P\}}(ceoRes)$ is unable to satisfy the goal belief $\mathscr{G}$, because neither thread Th_8 nor Th_9 contains the fact $\neg decr$ at time $t = 5$. Therefore we can disregard $Obs_{\{C,P\}}(ceoRes)$ as a solution candidate without evaluating the respective probabilities. Moreover, the solution candidate $Obs_{\{C,P\}}(contactG)$ does not induce any set of threads and can therefore be disregarded as well.

This results in the following hypothesis space for the stock market example:

[2] Again, we consider only the upper border u. Equivalent considerations hold for $\ell = 0$ and $\ell = u = 0$. Since such beliefs can be equivalently expressed as beliefs in the negated object $\neg\varphi$ (cf. the considerations on strict beliefs in Section 4.1), we do not consider these cases explicitly, but instead assume that the goal is rewritten accordingly.

$$\mathfrak{H} = \left\{ \begin{array}{l} \emptyset, \\ \{Obs_{\{C,P\}}(pr)\}, \\ \{Obs_{\{C,P\}}(pr), Obs_{\{C,P\}}(ceoRes)\}, \\ \{Obs_{\{C,P\}}(pr), Obs_{\{C,P\}}(contactG)\}, \\ \{Obs_{\{C,P\}}(pr), Obs_{\{C,P\}}(contactG), Obs_{\{C,P\}}(ceoRes)\} \end{array} \right\} \tag{6.20}$$

6.3. The Abduction Process

Using Definition 6.1 for the entailment of PDT Logic formulae, we can determine whether a candidate solution $\mathfrak{S} \in \mathfrak{H}$ is actually a solution to the abduction problem—i.e., $\mathfrak{S}$ together with the background knowledge $\mathbf{K}$ entails the goal $\mathscr{G}$—through corresponding satisfiability checks.

Checking satisfiability of a set of PDT Logic belief formulae $\mathfrak{B}$ with respect to a given set of threads $\hat{\mathcal{T}}$ is NP-complete (cf. Theorem 4.11 from Chapter 4). Building on this result, we obtain the following complexity result for deciding whether a solution exists for an instance of the PDT Logic abduction problem:

Theorem 6.1 (Complexity of PDT Abduction). *Let $\langle \mathbf{K}, H, \mathscr{G} \rangle$ be an instance of the PDT Logic abduction problem. Deciding whether a solution exists is Σ_2^P-complete.*

Proof. The proof works analogously to the proof of Theorem 4.2 in [MSS14]. Thus, here we only give a proof sketch adapted to belief formulae of PDT Logic. The complete proof from [MSS14] can be found in Appendix A.

Showing membership is straightforward: We can guess a potential solution $\mathfrak{S} \subseteq H$. Using Definition 6.1 and Theorem 4.11, it is easy to see that this solution can be verified in polynomial time by querying an NP oracle.

A known Σ_2^P–complete problem [SM73] is validity checking of a quantified Boolean formula Φ of the form $\exists X \forall Y \psi(X, Y)$ with mutually distinct Boolean variables $X = \langle x_1, ..., x_n \rangle$ and $Y = \langle y_1, ..., y_m \rangle$, respectively and $\psi(X, Y)$ a Boolean formula over the variables x_i and y_j. Intuitively, this problem has a close connection to the PDT Logic abduction problem, as we need to find some assignment to X (i.e., an abductive solution) such that the goal Y is always satisfied. Thus, we use the respective x_i as potential observation objects of the abduction problem, and set $\psi(X, Y)$ as the abduction goal; i.e., we do not restrict the set of possible threads, leave the set of belief formulae $\mathfrak{B}$ empty, pick an arbitrary agent a and define hypotheses and goal belief for this agent as follows:

$$\mathfrak{B} = \emptyset, \quad H = \bigcup_{i=1}^{n} \{Obs_{\{a\}}(x_i)_1, Obs_{\{a\}}(\neg x_i)_1\}, \quad \mathfrak{S} = B_{a,t}^{1,1}(\psi(X,Y))$$

Using this formulation, we can transform validity checks of any Boolean formula Φ of the above form to an instance of the PDT Logic abduction problem and thus show that the problem is Σ_2^P-hard. $\square$

In [MSS14], a geometric characterization for abduction in APT Logic is given. In a nutshell, for APT Logic the sets of probabilistic constraints $\mathfrak{B}$, potential solutions $\mathfrak{S} \subseteq \mathfrak{H}$ and goal $\mathscr{G}$ form convex polytopes. This allows for a geometric characterization of solutions to the abduction problem: $\mathfrak{S}$ is a solution if and only if the intersection of the polytopes induced by $\mathfrak{S}$ and $\mathfrak{B}$ is nonempty and is contained within the polytope induced by the goal $\mathscr{G}$. Unfortunately, this characterization is not applicable to PDT Logic. As already discussed in Section 4.3.3, due to the possibility of using disjunction and negation for belief formulae (cf. Definition 3.4 from Chapter 3), it can neither be guaranteed that a corresponding geometric representation results in a single polytope, nor that the resulting polytope(s) are convex. However, we can use the results on satisfiability checking in PDT Logic from Chapter 4 and adapt the approach from [MSS14] by substituting geometric polytope operations with according satisfiability checks. This leads to the identification of two distinct cases that will guide the abduction procedure:

Proposition 6.2. *Let $\langle \mathbf{K}, H, \mathscr{G} \rangle$ be an instance of the PDT Logic abduction problem and let $\mathfrak{S} \subseteq H$ be a potential solution to this problem. Then, the following observations hold for the abduction problem:*

1. *$\neg sat(\mathbf{K} \cup \{\mathscr{G}\})$, background knowledge and goal are inconsistent. Then, there is no solution to the abduction problem and no hypothesis $\mathfrak{S}$ has to be tested.*
2. *$\neg sat(\mathbf{K} \cup \mathfrak{S})$, the potential solution is inconsistent w.r.t. the background knowledge. Then, every potential solution $\mathfrak{S}'$ with $\mathfrak{S} \subseteq \mathfrak{S}' \subseteq H$ is also inconsistent, and therefore cannot be a solution to the abduction problem. Then, we can remove $\mathfrak{S}'$ from $\mathfrak{H}$ to prune the hypothesis space when searching for solutions to the abduction problem.*

The first check determines whether it is at all required to search for a solution to the abduction problem. The second case provides a pruning condition for the hypothesis search space $\mathfrak{H}$: if a solution candidate is not satisfiable together with the background knowledge, it is futile to test any superset of this solution. Using these properties, we obtain the abduction procedure depicted in Algorithm 2: after checking whether it is required to search for a solution at all (lines 2–3), the procedure iterates through all potential solutions from $\mathfrak{H}$, ordered by their respective size (lines 6–13), and prunes the

Algorithm 2 Abduction Algorithm for PDT Logic

```
 1: procedure ABDUCE($\mathbf{K}$,$H$,$\mathscr{G}$)
 2:     if $\neg sat(\mathbf{K} \cup \mathscr{G})$ then                    ▷ case 1: $\mathbf{K} \cup \mathscr{G}$ is inconsistent
 3:         return false
 4:     $\mathfrak{H} \leftarrow \{\mathfrak{S} \in 2^H : sp(\mathfrak{S})\}$      ▷ init search space as set of syntactically possible solutions
 5:     $i \leftarrow 0$
 6:     while ($\mathfrak{H} \neq \emptyset$ and $i \leq |H|$) do              ▷ test solutions in order of simplicity
 7:         for $\mathfrak{S} \in \mathfrak{H}$ with $|\mathfrak{S}| = i$ do
 8:             if $\neg sat(\mathbf{K} \cup \mathfrak{S} \cup \neg\mathscr{G})$ then           ▷ $\mathfrak{S}$ is a solution
 9:                 return $\mathfrak{S}$
10:             else
11:                 if $\neg sat(\mathbf{K} \cup \mathfrak{S})$ then     ▷ case 3: $\mathbf{K} \cup \mathfrak{S}$ is inconsistent, prune supersets
12:                     $\mathfrak{H} \leftarrow \mathfrak{H} \setminus \{\mathfrak{S}' : \mathfrak{S}' \in \mathfrak{H} \wedge \mathfrak{S}' \supseteq \mathfrak{S}\}$
13:         $i \leftarrow i + 1$
14:     return false
```

search space whenever some potential solution $\mathfrak{S}$ is inconsistent w.r.t. the background knowledge (line 12). The procedure terminates if a solution is found or the search space is empty.

Remark 6.2. [MSS14] provides two additional optimizations for abductive reasoning in APT Logic: First, it is checked whether $\neg sat(\mathbf{K} \cup \neg\mathscr{G})$ holds. In this case, every possible evolution induced by $\mathfrak{B}$ entails the goal $\mathscr{G}$ and thus the empty set $\emptyset$ is a solution to the abduction problem. These considerations cannot be transferred to PDT Logic directly: In general it might be possible that every thread induced by $\mathbf{K}$ indeed entails the goal $\mathscr{G}$. Also, it might be possible that the empty set $\emptyset$ is a solution to the abduction problem. However, for every nonempty hypothesis space $\mathfrak{H}$, these are two distinct cases: The former case considers *all* threads induced by $\mathbf{K}$, the latter only the subset of threads where no other hypothesis is chosen. This is best illustrated with the set of threads depicted in Figure 6.1: at time $t = 1$, P considers all threads possible, but even if no action (i.e., $\mathfrak{S} = \emptyset$) is taken, the set of threads that P considers possible is inevitably reduced to $\mathcal{K}^1_{P,2}$. Thus, opposed to the situation in APT Logic, the solution candidate $\emptyset$ is no special case and consequently we treat it just as any other hypothesis. The effect however remains the same in both formalisms: if $\emptyset$ solves the abduction problem, it is found first and returned as a solution. Second, another pruning condition is provided by arguing that for $\mathbf{K} \cup \mathfrak{S} \not\models \mathscr{G}$ (with $sat(\mathbf{K} \cup \mathfrak{S})$), any subset $\mathfrak{S}' \subseteq \mathfrak{S}$ cannot solve the abduction problem, either. This is not applicable in PDT Logic, because beliefs change with additional observations, and thus it is possible that $\mathfrak{S}'$ is indeed a solution to the abduction problem, while $\mathfrak{S}$ with additional observations is not. The ramifications of this condition are twofold: on the one hand, this additional condition enables pruning operations on the search space from both sides. While this can speed up the abduction procedure, this will not necessarily return minimal solutions.

Example 6.5 (stock markets continued)**.** We finish this example with an application of the abduction procedure given in Algorithm 2 to the example's hypothesis space as specified in Example 6.4. Recall that we aim at inducing the following goal (cf. (6.12)):

$$\mathscr{G} = B_{P,2}^{0.7,1}((\neg decr)_5) \equiv B_{P,2}^{0,0.3}((decr)_5)$$

To retrace the results of satisfiability checks, we can use the overview on P's potential beliefs given in Table 6.2. This leads to the following execution trace of Algorithm 2 applied to our example:

- First, it is checked whether $\neg sat(\mathbf{K} \cup \mathscr{G})$ holds (line 2). The last two entries in Table 6.2 show that the goal belief is induced in some threads. Thus, the goal is satisfiable with respect to the background knowledge and this condition evaluates to $false$.
- Then, the loop is entered and all solutions from $\mathfrak{H}$ are checked in order of increasing size. This leads exactly to the processing order of hypotheses and corresponding Kripke structures $\mathcal{K}_{P,2}^{j}$ as given in Table 6.2. Analyzing P's corresponding beliefs in the fact $decr_5$ shows that $\{Obs_{\{C,P\}}(pr), Obs_{\{C,P\}}(contactG)\}$ is the smallest solution to the abduction problem and accordingly, this is returned by the algorithm. Note that the superset $\{Obs_{\{C,P\}}(pr), Obs_{\{C,P\}}(contactG), Obs_{\{C,P\}}(ceoRes)\}$ also solves the problem, but as a smaller solution is identified before, this set is not checked.

Iterating through the search space in increasing order with respect to the solution size has to important consequences: First, it is ensured that any pruning operations due to inconsistent combinations of background knowledge and solution candidates are carried out as early as possible. The smaller the respective solution, the larger is the respective pruned superset and thus, pruning operations are applied most effectively. Second, any solution $\mathfrak{S}$ returned by Algorithm 2 is a minimal solution to the abduction problem.

Theorem 6.3. *Let $A = \langle \mathbf{K}, H, \mathscr{G} \rangle$ be an instance of the PDT Logic abduction problem. If A has a solution, then Algorithm 2 returns a minimal solution $\mathfrak{S}$ so that $\mathbf{K} \cup \mathfrak{S} \models \mathscr{G}$. Otherwise, the algorithm returns $false$.*

Proof. We start with showing that any set discarded in the pruning step (line 12) cannot be a solution to the abduction problem. If $\mathbf{K} \cup \mathfrak{S}$ is unsatisfiable, this set is already overly constrained so that no thread remains that could possibly satisfy all formulae in this set. Then, as observed in Proposition 6.2, adding further constraints will clearly still result in an empty set of possible threads. Thus, it is unnecessary to test any set $\mathfrak{S}' \supseteq \mathfrak{S}$ for possible solutions to the abduction problem.

If the abduction problem has a solution, it is clear that the loop in lines 6–13 will eventually find and return a solution, as all solution candidates are tested iteratively unless they are discarded as above. Since the algorithm iterates over the set of possible solutions by increasing size of the solution, any returned solution $\mathfrak{S}$ will necessarily be minimal. If there had been a smaller solution $\mathfrak{S}'$ with $|\mathfrak{S}'| < |\mathfrak{S}$, the algorithm would have terminated earlier by returning this solution $\mathfrak{S}'$. □

6.4. Concluding Remarks

In this chapter, we have shown how abduction can be formalized in the context of Probabilistic Doxastic Temporal (PDT) Logic. This provides means to formally reason about possible actions to induce a desired belief state for some agent. As shown with the example in this chapter, such kind of reasoning can be applied for example to identify suitable public relation strategies for market-listed companies. Deciding on suitable countermeasures in cyber security scenarios—as discussed in Chapter 5—is another example that can benefit from formalizing the abduction problem in PDT Logic.

Compared to abductive reasoning in other formalisms, the key feature of abduction in PDT Logic is the automatic identification and reduction of the hypothesis search space $\mathfrak{H}$. If background knowledge in the form of a set of threads $\hat{\mathcal{T}}$ together with a set of belief formulae $\mathfrak{B}$ or directly with priors $\mathcal{I}(\hat{\mathcal{T}})$ is given, all possible solutions to the abduction problem are represented through possible observations induced by this background knowledge. Based on the automatically constructed hypothesis space, we have developed a sound and complete algorithm to give a minimal solution to the abduction problem.

To the best of our knowledge, this is the first work that studies abduction in the context of dynamically evolving beliefs for multi-agent systems, and thus, the methods introduced in this chapter provide means for novel reasoning capabilities.

Chapter 7

Extending Time Frames to Markovian Streams

In its basic form, the definition of PDT Logic in Chapter 3 only allows for reasoning over finite time frames. In this chapter, we show how the temporal notion of finite time frames can be replaced with Markovian streams, i.e., under certain conditions we can represent domains with infinite streams of possible worlds. This provides means to integrate established stream reasoning systems (e.g., the well-known linear road benchmark [ACG$^+$04]) and probabilistic multi-agent belief operators into a coherent framework. This has two major advantages over finite time frames: First, during the modeling phase it is no longer necessary to fix a (rather arbitrary) maximum time point. If the selected time frame is too small, desired analysis results of the respective problem might become inaccessible, while an over-sized time frame will increase computational requirements unnecessarily. Second, a time frame with fixed start and end time points will require continuous re-initializations of the formalism. To illustrate this, consider the cyber security example from the previous chapter: The example always starts with a no-defense action at time point 0 and—if no attack is detected—continues with a further no-defense action at time point 3. As sequences of no observed attacks should be the prevailing state of the system, the formalism needs to be re-initialized frequently to correctly represent the current situation of the attack graph shown in Figure 5.1. Using a stream-based approach, the system could be continuously monitored and—if an attack is observed—the corresponding analysis can be carried out seamlessly without requiring a re-initialization of the formalism before. As long as no attack is observed, the analysis simply remains in the infinite sequence of alternating no-attack and no-defense nodes as already hinted in the center thread of Figure 5.1.

In the following, we show how threads in PDT Logic can be represented through infinite Markovian streams of possible worlds. Then, we show how finite-length time windows can be defined within those streams and provide adapted concepts of threads and prior probabilities to represent beliefs in stream-based models.

7.1. Substituting Threads with Infinite Streams of Possible Worlds

In Chapter 3, we originally defined threads as a mapping from a finite set of time points τ to a set of admissible worlds (cf. Definition 3.7). By replacing the finite set of time points τ with the set of natural numbers, we obtain the following definition of streams:

Definition 7.1 (Stream)**.** Let T be the set of natural numbers and let $\hat{\Omega}$ be the set of admissible worlds. Then, a *stream* St is a mapping

$$St : T \to \hat{\Omega}. \tag{7.1}$$

Since T ranges over the set of natural numbers, a stream is an infinite sequence of possible worlds and $St(t)$ identifies the actual world at time t according to stream St.

In order to provide a bridge between our previously introduced PDT semantics for finite-time models and the notion of infinite streams, we partition streams into fixed-length segments.

Definition 7.2 (Segment)**.** A *segment* S with fixed length s is a finite sequence of possible worlds:

$$S : \tau \to \hat{\Omega},\ \tau = \{1, ..., s\} \tag{7.2}$$

To identify the temporal position of specific segments, we enumerate segments with a parameter k, such that S_k represents the segment from time points $(k{\cdot}s)+1, ..., (k{\cdot}s)+s$, i.e., segment S_0 identifies the sequence of possible worlds from time $t = 1$ to $t = s$, segment S_1 identifies the sequence from $t = s + 1$ to $t = 2s$, and so on. This notion gives rises to an alternative representation of streams:

Corollary 7.1 (Stream segmentation)**.** *Every stream of possible worlds can be equivalently represented as a sequence of segments* $(S_k)_0^\infty$.

To distinguish the notions of time points in segments and streams, we call t an *absolute* time point if $t \in T$ identifies the actual time point in some stream St. We call t a *relative* time point if $t \in \tau$ identifies a time point within a specific segment without specifying the segment's position in the stream.

We assume that all possible streams of the application domain can be modeled with an arbitrarily large, but finite set of possible segments $\mathcal{S} = \{S^1, ..., S^n\}$. In most scenarios this requirement can be met through an appropriate modeling choice of the segment length s. In order to model the transitions from one segment to the next, we assume that the sequence of segments can be represented through an ergodic Markov chain [Nor98] according to the following definitions.

Definition 7.3 (Markov chain, adapted from [Nor98])**.** Let $\mathcal{S} = \{S^1, ..., S^n\}$ be a set of possible segments and let $\left(S^j_k\right)_0^\infty$ be a sequence of segments, such that all S^j_k are from $\mathcal{S}$. We say that $\left(S^j_k\right)_0^\infty$ is a *Markov chain* with initial distribution λ and transition matrix $M^{|\mathcal{S}|\times|\mathcal{S}|}$ if

1. S_0 has distribution λ;
2. for $k \geq 0$, conditional on $S_k = S^i_k$, S_{k+1} has distribution $(m_{ij} : j \in [1, ..., n])$ and is independent of $S_0, ...S_{k-1}$.

Property 1. of the above definition refers to the initial distribution of the segments, which we will determine below. Property 2. is the so-called Markov property and states that a stochastic process is *memoryless*, i.e., the transition probabilities to segment $k+1$ only depend on the current segment k. As M is a transition matrix, its columns each sum to one.

Definition 7.4 (Ergodicity)**.** A Markov chain with a set of segments $\mathcal{S}$ according to Definition 4.1 is *ergodic* if every segment $S^j \in \mathcal{S}$ is

1. *aperiodic*, i.e., returns to this segment can occur at irregular times, and
2. *positive recurrent*, i.e., every segment has a finite mean recurrence time.

In the following, we consider only streams of possible worlds that can be represented through an ergodic Markov chain according to Definitions 7.3 and 7.4. An important characteristic of an ergodic Markov chain is its steady state vector:

Lemma 7.2 (Stationary distribution, [Ser09])**.** *Any ergodic Markov chain with a set of segments $\mathcal{S}$ and a transition matrix $M^{|\mathcal{S}|\times|\mathcal{S}|}$ has a unique stationary distribution π (also called* steady state vector*), with the following properties:*

1. $0 < \pi_j < 1$,

2. $\displaystyle\sum_j^{|\mathcal{S}|} \pi_j = 1$,

3. $\displaystyle\pi_j = \sum_{i=1}^{|\mathcal{S}|} \pi_i m_{ij}$

The first two properties simply state that π is a probability distribution over the set of segments $\mathcal{S}$. The last property expresses that the chain converges to the steady state vector in the long run (regardless of the starting state). Thus, π_j specifies the long-term visiting rate of segment S^j.

7.2. Time Windows

In order to be able to reason about temporal changes in the near future, we use a time window W_{t,l_w} starting at absolute time t and having a length of l_w time points, i.e., some finite clipping of the infinite stream of possible worlds. We do not restrict the size of possible time windows in any way, and thus they may well span across multiple segments. In a sense, a time window is a myopic representation of the stream. This artificial limitation of time is meaningful to capture the influence of current observations over the probabilities of contemporary events: While the probabilities of all events in the remote future will converge to the long-term visiting rates of the respective segments in the Markov chain (cf. property 3. of Lemma 7.2), the probabilities of specific events within a certain time window may heavily depend on current observations and therefore significantly deviate from the long-term visiting rates.

Within a specific time window, we have fixed start and end time points, and thus, we can represent the different possible temporal evolutions within the time window (i.e., sequences of possible worlds for time $t, ..., t + l_w$) through threads as usual. Consequently, we use $Th_1, ..., Th_m$ to identify the threads within a certain time window. As usual, we use $Th(1), ..., Th(l_w)$ to identify the worlds of thread Th at the relative time points $1, ...l_w$. The possible threads within a time window are determined through the set of possible segments $\mathcal{S}$ and the transition matrix M of the Markov chain. Consequently, a thread in the time window W_{t,l_w} can be mapped onto a sequence of segments (each with length s) as

$$\begin{aligned} Th &\rightarrow S_k^{j_1}, S_{k+1}^{j_2}, ..., S_{k+x}^{j_x} \\ \text{with } k &= \lfloor (t/s) \rfloor, \\ \text{and } x &= \lfloor (t + l_w - 1)/s \rfloor, \end{aligned} \tag{7.3}$$

with $\lfloor x \rfloor$ being the floor operator, i.e., $\lfloor x \rfloor$ gives the largest integer previous to x, and $S^{j_1}, ..., S^{j_x}$ (not necessarily distinct) segments from $\mathcal{S}$. Note that time windows may be positioned arbitrarily, i.e., the start of a time window does not necessarily have to coincide with the start of a new segment, nor does the window length l_w have to be an integer multiple of the segment length s. That is, the sequence $S_k^{j_1}, S_{k+1}^{j_2}, ..., S_{k+x}^{j_x}$ from (7.3) may be longer than the actual time window thread Th. Specifically, the sequence may include additional time points $t - k, ..., t$ and $t + l_w - 1, ..., k + x + 1$. These additional time points are irrelevant to our analysis—this is only mentioned to stress the point that the position and size of time windows may be defined arbitrarily and entirely independent on the choice of segment length.

A visualization of the different temporal concepts is shown in Figure 7.1.

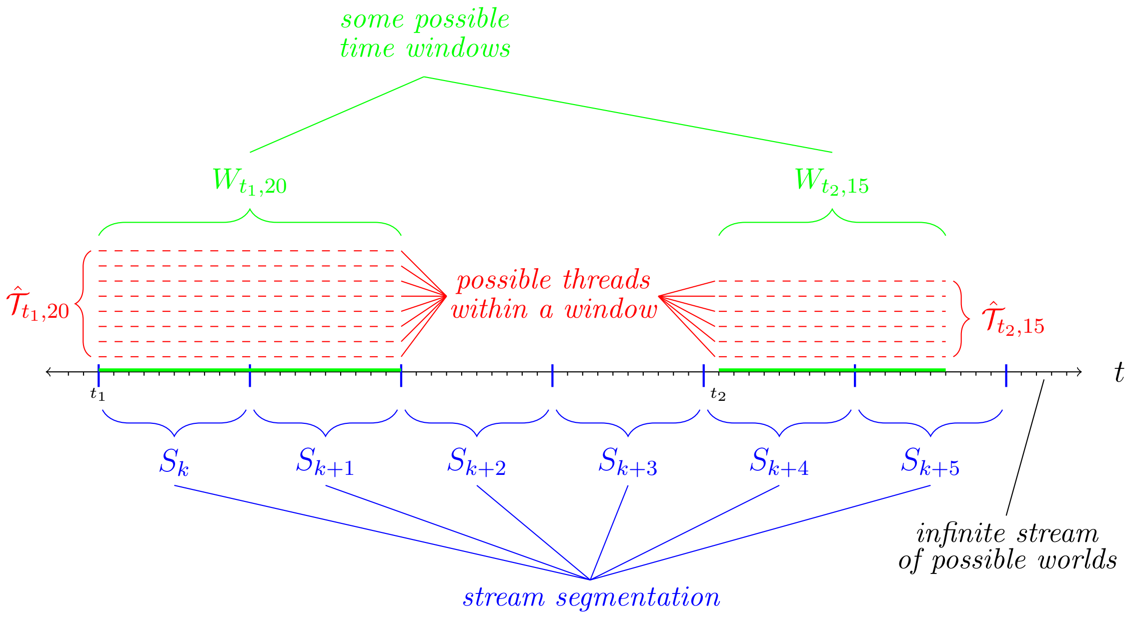

some possible time windows
$W_{t_1,20}$
$W_{t_2,15}$
possible threads within a window
$\hat{\mathcal{T}}_{t_1,20}$
$\hat{\mathcal{T}}_{t_2,15}$
t
t_1
t_2
S_k
S_{k+1}
S_{k+2}
S_{k+3}
S_{k+4}
S_{k+5}
infinite stream of possible worlds
stream segmentation

Figure 7.1.: Schematic depiction of the relations between different temporal concepts.

We resume the notation introduced in Chapter 3 and use $\hat{\mathcal{T}}_{t,l_w}$ to denote the set of all possible threads within a time window W_{t,l_w}. If the actual position and length of the time window are not important for our discussion, in the following we simplify notation and simply use $\hat{\mathcal{T}}$ to denote the set of threads to be analyzed. The set of possible threads within a time window can be determined as follows: Since the longterm visiting rates of all segments are nonzero (cf. Lemma 7.2), every possible segment $S \in \mathcal{S}$ can occur as the first segment $S_k^{j_1}$ in a thread $Th \in \hat{\mathcal{T}}_{t,l_w}$. Then, for every possible first segment $S_k^{j_1}$ in the time window, it is checked which of the entries $m_{j_1 j_2}$ in the corresponding row j_1 of the transition matrix are nonzero (a nonzero entry $m_{j_1 j_2}$ denotes a possible transition from segment S^{j_1} to segment S^{j_2}). All resulting combinations $S_k^{j_1}, S_{k+1}^{j_2}$ are then added as thread stubs to $\hat{\mathcal{T}}_{t,l_w}$. Continuing this for segments S_{k+1} to S_{k+x} yields the set of all possible threads $\hat{\mathcal{T}}_{t,l_w}$ in time window W_{t,l_w}.

7.3. Prior Probabilities for Threads within Time Windows

Now, with the concept of finite time windows W_{t,l_w} and a set of threads $\hat{\mathcal{T}}_{t,l_w}$ therein, we are close to the setting originally described in Section 3.2, where the semantics of PDT Logic based on finite threads was introduced. The only thing missing to complete the definition of PDT Logic for stream reasoning is a notion of prior probabilities for threads Th within time windows.

Since time windows can be placed at arbitrary positions within the stream, we can use the long-term visiting rates π together with the segment transition probabilities from M to determine the prior probability of a finite thread within the infinite stream of possible worlds:

Lemma 7.3. *Let $\mathcal{S}$ be a set of segments, Th be a thread mapped onto a sequence $S_k^{j_1}, S_{k+1}^{j_2}, ..., S_{k+x}^{j_x}$ of x of these segments, M be the transition matrix of the Markov process, and π be the corresponding steady state vector. Given that k is sufficiently large, the prior probability $\mathcal{I}(Th)$ of thread Th can be computed as*

$$\mathcal{I}(Th) = \pi_{j_1} \prod_{i=1}^{x-1} m_{j_i j_{i+1}} \tag{7.4}$$

Proof. For sufficiently large k, we can use the long-term visiting rates to determine probabilities. Then, the probability of starting the thread in segment S^{j_1} is given through element π_{j_1} of the steady state vector (cf. Lemma 7.2, property 3: the longterm visiting rate of S^{j_1} converges to π_{j_1}). From there, the probability of segment S^{j_2} occurring next is given through the entry $m_{j_1 j_2}$ in the transition matrix M of the Markov chain, and so on. Thus, the prior probability of thread Th can be expressed through the joint probability of respective segment sequence $S_k^{j_1}, S_{k+1}^{j_2}, ..., S_{k+x}^{j_x}$. □

Remark 7.1. Note that for a set of all possible threads within a time window (as discussed in the previous section), the corresponding priors as given in Lemma 7.3 sum to one. This follows immediately from the fact that both all elements in π and all rows in M sum to one, respectively. Thus, we can treat these priors exactly the same as in Chapter 3, without any need for normalization.

Up until now, we have still not specified an initial distribution λ for our Markov chain defined in Definition 7.3. Following the above considerations on the prior probabilities, it is useful to specify the initial distribution exactly as the stationary distribution of the Markov chain, i.e.,

$$\lambda = \pi. \tag{7.5}$$

Using this specification, time windows can be placed entirely arbitrarily, even at time points close to zero, i.e., Lemma 7.3 holds for all k, and the restriction on *sufficiently large k* can be dropped.

With these techniques, we have provided a bridge between the previously introduced semantics on finite time frames and infinite Markovian streams of possible worlds. After specifying a time window and determining the resulting set of possible threads with the respective priors, all methods introduced in the previous chapters can be applied to this time window without any modifications.

7.4. A Stream Example

To illustrate the application of PDT Logic for streams, we introduce a small example. This example is kept very simple to focus on the properties of Markovian streams of possible worlds and time windows therein.

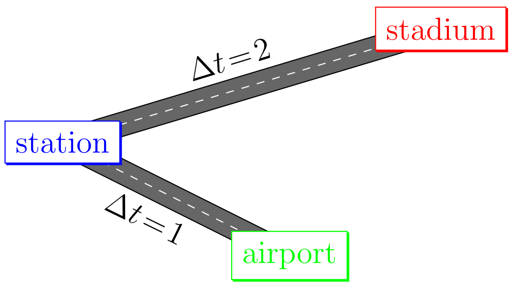
Assume that a taxi driver A is positioned at the central station and is occupied with driving two different routes from there, either to the airport (requiring one time step) or to the stadium (requiring two time steps). These routes of the taxi driver are depicted in Figure 7.2(a). Furthermore, we assume that the driver can take a break at the stadium every once in a while and inform her boss B about it. This means that both A and B know that A is at the stadium (and they are both aware that the other agent knows this), so we can model the effect of this as a shared observation $Obs_{\{A,B\}}(stadium)$. A possible segmentation of the infinite stream of possible worlds into segments could use segments of length three. This yields the set $\mathcal{S}$ of possible 3-step-sequences shown in Figure 7.2(b).[1] If we assume that driving to the airport has a probability of, say, 0.7, and the probability of taking a break while at the stadium is 0.3, we obtain the Markov chain depicted in Figure 7.2(c) with the corresponding transition matrix M shown in Figure 7.2(d).

One can easily verify that this yields the longterm visiting rates π_j of Segments S^j:

$$\pi \approx \begin{pmatrix} .172 & .074 & .245 & .150 & .105 & .105 & .032 & .074 & .045 \end{pmatrix} \tag{7.6}$$

To show the evolution of the agents' beliefs, consider a time window of length $l_w = 6$, i.e., a sequence of two segments, such that first time point of the time window coincides with the beginning of a new segment. Assume that we are interested in the beliefs of, say, A being at the stadium (i.e., $F = \mathit{stadium}$) at relative time $t = 4$, denoted by $Th(4) \models F$. Analysis of the set of segments $\mathcal{S}$ (Figure 7.2(b)) shows that $Th(4) \models F$ if and only if the second segment of the considered time window is S^5 or S^9. Thus, the probability of being at the stadium at relative time point 4 is equivalent to the property of segment S^5 or S^9 occurring as the second segment within the time window. From the transition model (Figure 7.2(c)) we get that this is possible if the first segment is S^2, S^7 (these enable transitions to S^5), or S^4 (the only segment with a transition to S^9). Hence, from the steady state vector (7.6) and the transition matrix from Figure 7.2(d) we obtain (through $\pi_2 \cdot m_{25} + \pi_7 \cdot m_{75} + \pi_4 \cdot m_{49} = 0.154$) that a correct prior belief of

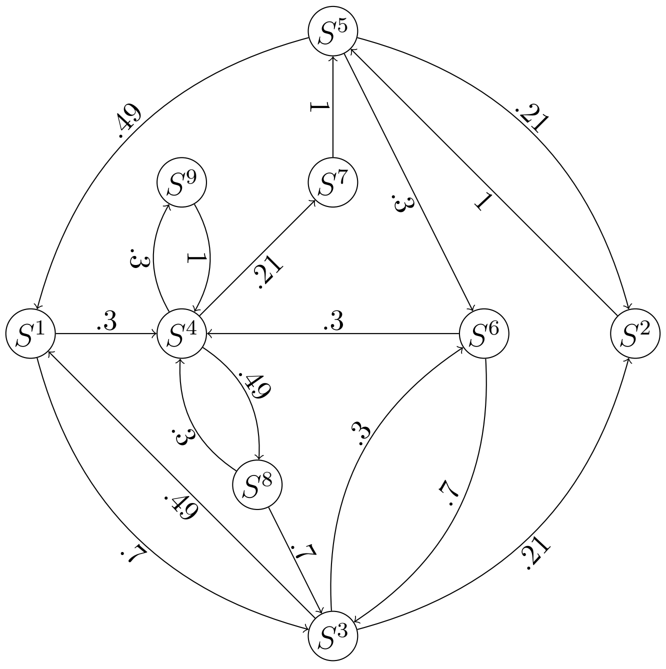
[1] For the sake of simplicity, we assume that a break always lasts exactly three time points. We could easily adapt the model to allow for breaks of arbitrary lengths by introducing additional segments with *break* prefixes for one or two time points. The general procedure remains the same, but as this blows up the segment space and thereby yields a more complex representation of the Markov chain, we refrain from doing so.



(a) possible routes

S^1:	airport	station	airport
S^2:	airport	station	enroute
S^3:	station	airport	station
S^4:	station	enroute	stadium
S^5:	stadium	enroute	station
S^6:	enroute	stadium	enroute
S^7:	enroute	station	enroute
S^8:	enroute	station	airport
S^9:	stadium	stadium ($Obs_{\{A,B\}}(stadium)$)	stadium
t	1	2	3

(b) the set $\mathcal{S}$ of possible 3-step sequences



(c) segment transitions

$$M = \begin{pmatrix} 0 & 0 & .7 & .3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ .49 & .21 & 0 & 0 & 0 & .3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & .21 & .49 & .3 \\ .49 & .21 & 0 & 0 & 0 & .3 & 0 & 0 & 0 \\ 0 & 0 & .7 & .3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & .7 & .3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(d) transition matrix

Figure 7.2.: Taxi driver example: (a) depicts the possible routes, (b) all possible 3-step sequences, (c) the resulting possible segment transitions, and (d) the corresponding transition matrix.

both agents in this event can be specified as

$$B_{i,0}^{0.15,0.16}(F_4),\ i \in \{A, B\}. \tag{7.7}$$

Now suppose that time evolves for 3 steps and that the actual events are the ones depicted in S^4 (i.e., A is actually at the stadium at time $t = 3$). Then, driver A can update her belief in the event F_4 to

$$B_{A,3}^{0.3\ 0.3}(F_4),$$

because if she is already to the stadium at $t = 3$, she will still be there at $t = 4$ if and only if she takes a break (this corresponds to the transition from segment S^4 to S^9). while B (lacking any new information) maintains the same belief expressed at time 0. Since A is aware that B did not receive new information,

$$B_{A,3}^{1,1}(B_{B,3}^{0.15\ 0.16}(F_4))$$

also holds.

Finally, consider the situation at $t = 4$: if A decides to take a break, she informs B (represented through the shared observation $Obs_{\{A,B\}}(stadium)$ in S^9), and consequently,

$$B_{i,4}^{1,1}(F_4),\ i \in A, B$$

holds. Else, if A does not take a break, her beliefs are updated to

$$B_{A,4}^{0,0}(F_4),$$

while the following expression holds for B's updated beliefs (through elimination of possibility S^9)

$$B_{B,4}^{0.10,0.11}(F_4).$$

This small example shows how we can model a problem in PDT Logic with infinite streams of possible worlds. Within such a stream, we can place finite length time windows and then—as discussed in previous chapters—carry out an analysis of evolving multi-agent beliefs. Again, this example shows how the individual beliefs of agents can diverge over time depending on their respective information.

7.5. Concluding Remarks

By extending PDT Logic to Markovian streams of possible worlds, we can represent infinite time domains. This results in a general framework to reason about belief change in multi-agent stream scenarios. Although the actual reasoning about temporal beliefs is carried out for finite windows of time, arbitrary placements of these windows within infinite streams of possible worlds enable unlimited temporal reasoning in stream domains. For example, one could employ a sliding window mechanism to provide for continuous reasoning about the evolution of beliefs. As described in the introductory example, using an infinite stream of possible worlds as an input to the formalism can be beneficial because it allows for a continuing analysis of changing situations without requiring frequent re-initializations. While the employed model of infinite streams of possible worlds based on a Markov chain restricts the application domains to scenarios that can be modeled through recurring segments, this still allows for a wide range of applications because the properties of most domains can be captured with a suitable selection of length and number of segments.

Chapter 8

Conclusion

In this work, by extending APT Logic to dynamic scenarios with multiple agents, we have developed a general framework to represent and reason about belief change in multi-agent systems. Apart from lifting the single-agent case of APT Logic to multiple agents, we have also provided a suitable semantics to the temporal evolution of beliefs. The resulting framework extends previous work on dynamic multi-agent epistemic logics by enabling the quantification of agents' beliefs through imprecise probabilities. An explicit notion of temporal relationships is provided through temporal rules building on the concept of frequency functions.

Probabilistic information about a problem domain can be specified through the knowledge of domain experts or can be gathered through suitable data analysis methods. The quantification of beliefs with imprecise probabilities instead of precise values has the advantage that when modeling the problem, it is not only possible to provide probabilistic background knowledge about the problem domain, but the certainty of the respective specifications can be specified, as well. Narrow interval quantifications reflect a high certainty and vice versa. This can be a significant advantage compared to other probabilistic approaches, because in most approaches, sharp probability values are required, which a human can usually not express with precise values and thus humans basically have to rely on guesses for these values. Enforcing precise values—in particular if these values are determined by guesses—can yield misleading results. PDT Logic is not exposed to this problem, because specifying sharp values is not required to model a problem.

We have shown that there are alternative ways of specifying problems in PDT Logic, either through explicit enumerations of possible threads—optionally with according prior probabilities—or through a set of appropriate rules. Each approach exhibits its specific advantages and drawbacks: For many problem domains, requiring an exhaustive enumeration of all possible threads and according prior probabilities poses a severe obstacle for modeling the respective scenarios, as such a specification is practically unmanageable. On the other hand, there are problem domains (e.g., attack graphs in cyber security scenarios) that come with such an explicit specification anyways. For these types of problems, we have shown that it is possible to check satisfiability of these models very efficiently.

To overcome the modeling disadvantages of the exhaustive thread representation, we have also shown how a problem domain can be solely specified through a set of PDT Logic belief formulae. For most problem domains, this is a more natural way of specifying the problem. Also, this provides means to easily adapt many existing problems—that are specified in other formal languages as sets of rules—to PDT Logic. On the other hand, waiving the requirement of manually specified possible threads significantly hardens the problem of checking satisfiability of a set of PDT Logic formulae. This is because it is not possible to simply construct a single thread that satisfies all formulae, but instead the interplay of all potential threads and their respective probability assignments has to be considered.

To illustrate potential applications of PDT Logic, we have discussed two application scenarios in detail, namely the application to cyber security threat analysis and the interaction strategies between companies and market participants. Naturally, a theoretic discussion of these examples inevitably requires the use of strongly simplified problem domains in order to keep the examples' sizes manageable. Nevertheless, these examples indicate practical application scenarios where formal models based PDT Logic yield additional benefits over existing methodologies.

The core of this thesis consists of the development of suitable syntax and semantics of PDT Logic, together with alternative formal problem specification approaches and associated decision procedures. In additional chapters we have already provided some extensions to the core formalism. First, we have shown how abductive reasoning can be performed in PDT Logic. This enables the formal specification of desired goal belief states for some agents. Then, potential measures to induce this desired goal state can be identified automatically and—if the problem is solvable—a minimal solution to the abduction problem will be returned. Second, we have shown how finite time frames of the core formalism can be replaced with infinite Markovian streams of possible worlds. By placing time windows within these streams and adapting the prior probability definitions, we have shown that PDT Logic can be applied to problem domains with infinite models of time as well.

Based on search heuristics, in this thesis we have outlined a general procedure to decide satisfiability of a given set of PDT Logic belief formulae. A first step for future work could be the adoption of existing optimization techniques from research on logic reasoners to implement an actual reasoner to employ PDT Logic for practical domains.

Furthermore, the formalism developed in this thesis can be used as a foundation to develop frameworks that reason about actions and strategies. Various works (e.g., [JÅ07], [BG11], [vdHW03]) exist that combine epistemic and doxastic logics with a formal representation of actions and utilities, such that they enable reasoning about optimal strategies to maximize expected utilities of an agent. Using PDT Logic as a formal foundation can enhance these approaches by enabling strategies based on imprecise beliefs, i.e., based on their current belief state, agents can then reason about potential strategies such that they induce states which maximize their respective utility functions. The utility function of an agent in turn can both be based on actual states of the world as well as on the belief state of the respective agent or other agents.

Appendix A

Complexity Proof for Abduction in APT Logic

In this appendix we replicate the complexity result from [SPSS11] for abduction in APT Logic. Analogously to this proof, we can derive complexity results for abduction in PDT Logic (cf. Theorem 6.1).

Before quoting the actual results, we briefly introduce some preliminaries from [SPSS11] that are used in the following proof:

- An abduction problem in APT Logic is specified as $P = \langle \Pi, H, g \rangle$ with an APT Logic program Π, which induces a set of linear constraints, a hypothesis space H and a desired goal g.
- I is a temporal probabilistic interpretation (also referred to as *tp interpretation*). This concept is equivalent to prior interpretations in PDT Logic.
- The concept of threads Th is equivalent to the thread-concept used in PDT Logic. (the concept of actual worlds within threads at specific time points differs between PDT Logic and APT Logic, but this is irrelevant for the following proof.)
- Theorem 4.1 in [SPSS11] states that consistency checking for an APT Logic program Π is NP-complete.

The remainder of this appendix is a verbatim reproduction from [SPSS11].

Theorem 4.2 (Solution existence)**.** *Let $P = \langle \Pi, H, g \rangle$ be an instance of the APT abduction problem. Deciding whether a solution exists for P is Σ_2^P-complete. Σ_2^P-hardness holds even if Π is empty.*

Proof. (Membership) We guess $S \subseteq H$ and verify that S is a solution by checking whether $\Pi \cup S$ is consistent and $\Pi \cup S \models g$ (it follows from Theorem 4.1 in [SPSS11] that S can be verified in polynomial time by querying an NP oracle).

(Hardness) We reduce the problem of checking whether a quantified Boolean formula Φ of the form $\exists X \forall Y \psi(X, Y)$ is valid to our problem—here $X = \langle x_1, ..., x_n \rangle$ and $Y = \langle y_1, ..., y_m \rangle$ are tuples of mutually distinct propositional variables and $\psi(X, Y)$ is a propositional formula over the x_i's and the y_j's. The aforementioned problem is Σ_2^P-

complete [SM73]. We construct an instance $P = \langle \Pi, H, g \rangle$ of the APT abduction problem as follows:

$$\Pi = \emptyset$$
$$H = \bigcup_{i=1}^{n} \{x_i : [1,1,1], \neg x_i : [1,1,1]\}$$
$$g = \psi(X, Y) : [1,1,1].$$

Clearly, the reduction can be accomplished in polynomial time. In the following, with a slight abuse of notation, we treat X and Y as sets. We now show that Φ is valid if and only if there exists a solution for P.

($\Rightarrow$) Suppose Φ is valid. Since Φ is valid, then there exists a truth assignment $\phi : X \to \{true, false\}$ such that $\forall Y \psi'(Y)$ is valid, where $\psi'(Y)$ is the propositional formula obtained from $\psi(X, Y)$ by replacing each occurrence of x_i with $\phi(x_i)$, for all $x_i \in X$. Let $S = \{x_i : [1,1,1] | x_i \in X \wedge \phi(x_i) = true\} \cup \{\neg x_i : [1,1,1] | x_i \in X \wedge \phi(x_i) = false\}$. We show that S is a solution of P. Obviously, $S \subseteq H$. Moreover, $\Pi \cup S$ is consistent. To see this, consider a thread Th s.t. $Th(1) = \{x_i | x_i \in X \wedge \phi(x_i) = true\}$, and let I be a tp interpretation s.t. $I(Th) = 1$. It is easy to see that $I \models \Pi \cup S$. It remains to show that $\Pi \cup S \models g$; that is, every tp interpretation satisfying $\Pi \cup S$ satisfies g as well. Assume I is a tp interpretation s.t. $I \models \Pi \cup S$. Let f be the propositional formula $\bigwedge_{x_i \in X, \phi(x_i)=true} x_i \wedge \bigwedge_{x_i \in X, \phi(x_i)=false} \neg x_i$. Since $I \models S$, then $\sum_{Th \in \mathcal{T}, Th(1) \models f} I(Th) = 1$. To prove the latter statement, suppose by contradiction that it does not hold; that is, there exists a thread $Th' \in \mathcal{T}$ s.t. $Th'(1) \not\models f$ and $I(Th') > 0$. it is easy to check that $I \not\models S$, which is a contradiction. Note that, for any thread Th, if $Th(1) \models f$, then $Th(1) \models \psi(X, Y)$. Hence, $\sum_{Th \in \mathcal{T}, Th(1) \models \psi(X,Y)} I(Th) = 1$, that is $I \models g$.

($\Leftarrow$) Suppose there exists a solution S for P. Let $X' \subseteq X$ be the set of variables x_i in X s.t. either $x_i : [1,1,1] \in S$ or $\neg x_i : [1,1,1] \in S$ (note that both cannot hold in order for S to be consistent). We first show that $S' = S \cup \bigcup_{x_i \in X - X'} \{x_i : [1,1,1]\}$ is a solution as well. It is obvious that $S' \subseteq H$. Moreover, $\Pi \cup S'$ is consistent. To see this, consider a thread Th s.t. $Th(1) = \{x_i | x_i : [1,1,1] \in S'\}$, and let I be a tp interpretation s.t. $I(Th) = 1$. It is easy to see that $I \models \Pi \cup S'$. We now show that $\Pi \cup S' \models g$. Since $S \subseteq S'$, then the set of tp interpretations that satisfy $\Pi \cup S'$ is a subset of the tp interpretations satisfying $\Pi \cup S$. As the latter ones satisfy g, then $\Pi \cup S' \models g$. Hence, S' is a solution.

We now show that Φ is valid. Let $\phi : X \to \{true, false\}$ be a truth assignment such that $\phi(x_i) = true$ if and only if $x_i : [1,1,1] \in S'$ and $\phi(x_i) = false$ if and only if $\neg x_i : [1,1,1] \in S'$, for all $x_i \in X$. We show that $\forall Y \psi'(Y)$ is valid, where $\psi'(Y)$ is the propositional formula obtained from $\psi(X, Y)$ by replacing each occurrence of x_i with $\phi(x_i)$, for all $x_i \in X$. Suppose by contradiction that $\forall Y \psi'(Y)$ is not valid. Then, there exists a truth assignment $\phi' : Y \rightarrow \{true, false\}$ that does not satisfy $\psi'(Y)$. Consider a thread Th s.t. $Th(1) = \{x_i | x_i \in X \wedge \phi(x_i) = true\} \cup \{y_i | y_i \in Y \wedge \phi'(y_i) = true\}$, and let I be a tp interpretation s.t. $I(Th) = 1$. It is easy to check that $I \models \Pi \cup S'$ and $I \not\models g$, which is a contradiction.

Not that the previous reduction uses an empty program Π. $\square$

List of Figures

3.1. Train example threads 18
3.2. Frequency functions example thread 23

4.1. Taxonomy of belief formulae 64
4.2. Initial representative thread generation 78
4.3. Application of ple-expansions 83
4.4. Continued application of ple-expansions 84

5.1. Cyber security example threads 93

6.1. Abduction example threads 103

7.1. Visualization of temporal concepts 117
7.2. Stream reasoning example 120

Bibliography

[ACG+04] A. Arasu, M. Cherniack, E. Galvez, D. Maier, A. S. Maskey, E. Ryvkina, M. Stonebraker, and R. Tibbetts. Linear Road: A Stream Data Management Benchmark. In *Proceedings of the Thirtieth International Conference on Very Large Data Bases*, VLDB 04, 2004.

[Aum76] Robert J. Aumann. Agreeing to Disagree. *The Annals of Statistics*, 4(6):1236–1239, November 1976.

[Bal85] Egon Balas. Disjunctive Programming and a Hierarchy of Relaxations for Discrete Optimization Problems. *SIAM Journal on Algebraic Discrete Methods*, 6(3):466–486, July 1985.

[Bal98] Egon Balas. Disjunctive Programming: Properties of the Convex Hull of Feasible Points. *Discrete Applied Mathematics*, 89(1):3–44, December 1998.

[Bar00] Chitta Baral. Abductive Reasoning through Filtering. *Artificial Intelligence*, 120(1):1–28, June 2000.

[BCC93] Egon Balas, Sebastián Ceria, and Gérard Cornuéjols. A Lift-and-project Cutting Plane Algorithm for Mixed 0-1 Programs. *Mathematical Programming*, 58(3):295–324, February 1993.

[BCC96] Egon Balas, Sebastián Ceria, and Gérard Cornuéjols. Mixed 0-1 Programming by Lift-and-project in a Branch-and-cut Framework. *Management Science*, 42(9):1229–1246, September 1996.

[BEL+01] Matthias Baaz, Uwe Egly, Alexander Leitsch, Jean Goubault-Larrecq, and David Plaisted. Normal Form Transformations. In Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, chapter 5, pages 273 – 333. MIT Press, 2001.

[BFL07] Livio Bertacco, Matteo Fischetti, and Andrea Lodi. A Feasibility Pump Heuristic for general Mixed-Integer Problems. *Discrete Optimization*, 4(1):63–76, March 2007.

[BG11] Chitta Baral and Gregory Gelfond. On representing actions in multi-agent domains. In Marcello Balduccini and Tran Cao Son, editors, *Logic Programming, Knowledge Representation, and Nonmonotonic Reasoning*, pages 213–232. Springer, 2011.

[Bie96] Daniel Bienstock. Computational Study of a Family of Mixed-Integer Quadratic Programming Problems. *Mathematical Programming*, 74(2):121–140, August 1996.

[BM04] Alexandru Baltag and Lawrence S. Moss. Logics for Epistemic Programs. *Synthese*, 139(2):165–224, March 2004.

[BMS98] Alexandru Baltag, Lawrence S. Moss, and Slawomir Solecki. The Logic of Public Announcements, Common Knowledge, and Private Suspicions. In *Proceedings of the Seventh Conference on Theoretical Aspects of Rationality and Knowledge*, TARK 98, pages 43–56, 1998.

[BP02] Egon Balas and Michael Perregaard. Lift-and-project for Mixed 0-1 Programming: Recent Progress. *Discrete Applied Mathematics*, 123(1):129–154, November 2002.

[Bra15] Seamus Bradley. Imprecise probabilities. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2015 edition, 2015.

[CEMS08] Martin W. Cripps, Jeffrey C. Ely, George J. Mailath, and Larry Samuelson. Common Learning. *Econometrica*, 76(4):909–933, July 2008.

[Com] The Computational Infrastructure For Operations Research (COIN-OR) Project. CBC (Coin-or branch and cut) user guide. `http://www.coin-or.org/Cbc/index.html`. accessed: 2016-04-15.

[Coo71] Stephen A. Cook. The Complexity of Theorem-proving Procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC 71, 1971.

[dCFFvdH08] Nivea de Carvalho Ferreira, Michael Fisher, and Wiebe van der Hoek. Specifying and Reasoning about Uncertain Agents. *International Journal of Approximate Reasoning*, 49(1):35–51, September 2008.

[DPDPR09] Alessandro Dal Palù, Agostino Dovier, Enrico Pontelli, and Gianfranco Rossi. Gasp: Answer set programming with lazy grounding. *Fundamenta Informaticae - Advances in Computational Logic*, 96(3):297–322, August 2009.

[EG95] Thomas Eiter and Georg Gottlob. The Complexity of Logic-based Abduction. *Journal of the ACM*, 42(1):3–42, January 1995.

[Ell61] Daniel Ellsberg. Risk, Ambiguity, and the Savage Axioms. *The Quar-*

terly Journal of Economics, 75(4):643–669, November 1961.

[Etc99] John Etchemendy. Logical consequence. In Robert Audi, editor, *The Cambridge Dictionary of Philosophy, 2nd edition.* Cambridge University Press, 1999.

[FGL05] Matteo Fischetti, Fred Glover, and Andrea Lodi. The Feasibility Pump. *Mathematical Programming*, 104(1):91–104, September 2005.

[FH94] Ronald Fagin and Joseph Y. Halpern. Reasoning about Knowledge and Probability. *Journal of the ACM*, 41(2):340–367, March 1994.

[FHMV95] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge.* MIT Press, 1995.

[FLP12] Wolfgang Faber, Nicola Leone, and Simona Perri. The intelligent grounder of DLV. In *Correct Reasoning: Essays on Logic-Based AI in Honour of Vladimir Lifschitz*, pages 247–264. Springer, 2012.

[GG97] Jelle Gerbrandy and Willem Groeneveld. Reasoning About Information Change. *Journal of Logic, Language and Information*, 6(2):147–169, April 1997.

[GH03] Peter D. Grünwald and Joseph Y. Halpern. Updating Probabilities. *Journal of Artificial Intelligence Research*, 19(1):243–278, July 2003.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., 1979.

[Gnu] The Gnu Project. GLPK: GNU Linear Programming Kit. `http://www.gnu.org/software/glpk/glpk.html`. accessed: 2016-04-15.

[Gur] Gurobi Optimization, Inc. Gurobi optimizer reference manual. `http://www.gurobi.com/documentation/`. accessed: 2016-04-15.

[Har67] John C. Harsanyi. Games with Incomplete Information Played by 'Bayesian' Players. Part I. The Basic Model. *Management Science*, 14(3):159–182, November 1967.

[Har68a] John C. Harsanyi. Games with Incomplete Information Played by 'Bayesian' Players. Part II. Bayesian Equilibrium Points. *Management Science*, 14(5):320–324, January 1968.

[Har68b] John C. Harsanyi. Games with Incomplete Information Played by 'Bayesian' Players. Part III. The Basic Probability Distribution of the Game. *Management Science*, 14(7):486–502, March 1968.

[Hin62] Jaakko Hintikka. *Knowledge and Belief: An Introduction to the Logic* *of the Two Notions*. Cornell University Press, 1962.

[HSS09] Joseph Y. Halpern, Dov Samet, and Ella Segev. Defining Knowledge in Terms of Belief: The Modal Logic Perspective. *The Review of Symbolic Logic*, 2(3):469–487, September 2009.

[ILO] IBM ILOG. CPLEX Optimizer. `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/`. accessed: 2016-04-15.

[ILP06] Kyle Ingols, Richard Lippmann, and Keith Piwowarski. Practical Attack Graph Generation for Network Defense. In *Proceedings of the Twenty-second Annual Computer Security Applications Conference*, ACSAC 06. IEEE Computer Society, 2006.

[JÅ07] Wojciech Jamroga and Thomas Ågotnes. Constructive Knowledge: What Agents can achieve under Imperfect Information. *Journal of Applied Non-Classical Logics*, 17(4):423–475, 2007.

[JJ96] John R. Josephson and Susan G. Josephson, editors. *Abductive Inference: Computation, Philosophy, Technology.* Cambridge University Press, 1996.

[Jos96] John R. Josephson. Conceptual Analysis of Abduction. In John R. Josephson and Susan G. Josephson, editors, *Abductive Inference: Computation, Philosophy, Technology*, chapter 1, pages 5–30. Cambridge University Press, 1996.

[Koo03] Barteld P. Kooi. Probabilistic Dynamic Epistemic Logic. *Journal of Logic, Language and Information*, 12(4):381–408, September 2003.

[Kri63] Saul A. Kripke. Semantical Considerations on Modal Logic. *Acta Philosophica Fennica*, 16:83–94, 1963.

[LIS$^+$06] Richard Lippmann, Kyle Ingols, Chris Scott, Keith Piwowarski, Kendra Kratkiewicz, Mike Artz, and Robert Cunningham. Validating and Restoring Defense in Depth using Attack Graphs. In *Military Communications Conference*, MILCOM 06. IEEE Computer Society, 2006.

[Llo87] John W. Lloyd. *Foundations of Logic Programming, 2nd Edition.* Springer, 1987.

[MK00] Brian Milch and Daphne Koller. Probabilistic Models for Agent's Beliefs and Decisions. In *Proceedings of the Sixteenth Annual Conference on Uncertainty in Artificial Intelligence*, UAI 00. Morgan Kaufmann Publishers Inc., 2000.

[MSS14] Cristian Molinaro, Amy Sliva, and V. S. Subrahmanian. Super-Solutions: Succinctly Representing Solutions in Abductive Annotated

Probabilistic Temporal Logic. *ACM Transactions on Computational Logic*, 15(3):18:1–18:35, July 2014.

[Mur83] Katta G. Murty. *Linear Programming.* John Wiley & Sons, 1983.

[NJ08] Steven Noel and Sushil Jajodia. Optimal IDS Sensor Placement and Alert Prioritization Using Attack Graphs. *Journal of Network and Systems Management*, 16(3):259–275, September 2008.

[Nor98] James R. Norris. *Markov Chains.* Cambridge University Press, 1998.

[OGA05] Xinming Ou, Sudhakar Govindavajhala, and Andrew W Appel. MulVAL: A Logic-based Network Security Analyzer. In *Proceedings of the Fourteenth Conference on USENIX Security Symposium*, SSYM 05. USENIX Association, 2005.

[Pla89] Jan Plaza. Logics of public communications. In *Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems: Poster session program*, ISMIS 89. Oak Ridge National Laboratory, 1989.

[Pla07] Jan Plaza. Logics of Public Communications. *Synthese*, 158(2):165–179, September 2007.

[Poo97] David Poole. The Independent Choice Logic for Modelling Multiple Agents under Uncertainty. *Artificial Intelligence*, 94(1):7 – 56, July 1997.

[PR03] Rohit Parikh and Ramaswamy Ramanujam. A Knowledge Based Semantics of Messages. *Journal of Logic, Language and Information*, 12(4):453–467, September 2003.

[PS98] Cynthia Phillips and Laura Painton Swiler. A Graph-based System for Network-vulnerability Analysis. In *Proceedings of the 1998 Workshop on New security paradigms*, NSPW 98. ACM, 1998.

[Rei01] Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems.* MIT Press, 2001.

[RKT10] Arpan Roy, Dong Seong Kim, and Kishor S Trivedi. Cyber Security Analysis using Attack Countermeasure Trees. In *Proceedigns of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*, CSIIRW 10. ACM, 2010.

[RKT12] Arpan Roy, Dong Seong Kim, and Kishor S Trivedi. Scalable Optimal Countermeasure Selection using Implicit Enumeration on Attack Countermeasure Trees. In *Proceedings of the Forty-second Annual International Conference on Dependable Systems and Networks*, DSN 12.

IEEE, 2012.

[Sac08] Joshua Sack. Temporal Languages for Epistemic Programs. *Journal of Logic, Language and Information*, 17(2):183–216, April 2008.

[Sac09] Joshua Sack. Extending Probabilistic Dynamic Epistemic Logic. *Synthese*, 169(2):241–257, July 2009.

[Sch86] Alexander Schrijver. *Theory of Linear and Integer Programming.* John Wiley & Sons, 1986.

[Ser09] Richard Serfozo. *Basics of Applied Stochastic Processes.* Springer, 2009.

[SLB09] Yoav Shoham and Kevin Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations.* Cambridge University Press, 2009.

[SM73] L. J. Stockmeyer and A. R. Meyer. Word Problems Requiring Exponential Time(Preliminary Report). In *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*, STOC 73. ACM, 1973.

[SPSS11] Paulo Shakarian, Austin Parker, Gerardo Simari, and Venkatramana S. Subrahmanian. Annotated Probabilistic Temporal Logic. *ACM Transactions on Computational Logic*, 12(2):14:1–14:44, January 2011.

[SSS12] Paulo Shakarian, Gerardo I. Simari, and V. S. Subrahmanian. Annotated Probabilistic Temporal Logic: Approximate Fixpoint Implementation. *ACM Transactions on Computational Logic*, 13(2):13:1–13:33, April 2012.

[vB03] Johan van Benthem. Conditional Probability Meets Update Logic. *Journal of Logic, Language and Information*, 12(4):409–421, September 2003.

[vBGHP09] Johan van Benthem, Jelle Gerbrandy, Tomohiro Hoshi, and Eric Pacuit. Merging Frameworks for Interaction. *Journal of Philosophical Logic*, 38(5):491–526, October 2009.

[vBGK09] Johan van Benthem, Jelle Gerbrandy, and Barteld Kooi. Dynamic Update with Probabilities. *Studia Logica*, 93(1):67–96, October 2009.

[vdH97] Wiebe van der Hoek. Some Considerations on the Logic PFD: A Logic combining Modality and Probability. *Journal of Applied Non-Classical Logics*, 7(3):287–307, 1997.

[vdHW03] Wiebe van der Hoek and Michael Woolridge. Cooperation, Knowledge, and Time: Alternating-time Temporal Epistemic Logic and its Applications. *Studia Logica*, 75(1):125–157, October 2003.

[vDvdHK07] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi. *Dynamic Epistemic Logic.* Springer, 2007.

[vE14] Jan van Eijck. Dynamic epistemic logics. In *Johan van Benthem on Logical and Informational Dynamics*, chapter 7, pages 175–202. Springer, 2014.

[vES14] Jan van Eijck and François Schwarzentruber. Epistemic Probability Logic Simplified. In Rajeev Goré, Barteld P. Kooi, and Agi Kurucz, editors, *Advances in Modal Logic 10, invited and contributed papers from the tenth conference on "Advances in Modal Logic,"*, AiML 14. College Publications, 2014.

[vS90] Marilyn vos Savant. Ask Marilyn. *Parade Magazine*, 16, September 9, 1990.

[Wil09] H. Paul Williams. *Logic and Integer Programming.* Springer, 2009.