UNIVERSIDADE DA CORUÑA

COMPUTER SCIENCE DEPARTMENT

Temporal Answer Set Programming

PHD THESIS

Martín Diéguez Lodeiro

2015

COMPUTER SCIENCE DEPARTMENT

Temporal Answer Set Programming

PHD THESIS

Martín Diéguez Lodeiro

PhD Supervisor:
Dr. Pedro Cabalar Fernández

2015

UNIVERSIDADE DA CORUÑA

PhD Thesis

Temporal Answer Set Programming

Martín Diéguez Lodeiro
PhD Supervisor:
Dr. Pedro Cabalar Fernández

Thesis committee:
Dr. David Pearce (chair)
Dr. Philippe Balbiani
Dr. Stéphane Demri
Dr. Agustín Valverde Ramos
Dr. Concepción Vidal Martín

Substitute members:
Dr. Stefania Costantini
Dr. Manuel Ojeda Aciego

D. Pedro Cabalar Fernández, Profesor Titular de Universidade na área de Ciencias da Computación e Intelixencia Artificial da Universidade da Coruña

CERTIFICA

Que a presente memoria titulada **Temporal Answer Set Programming** foi realizada baixo a súa dirección e constitúe a Tese que presenta **Martín Diéguez Lodeiro** para optar ao grao de Doutor pola Universidade da Coruña.

Asina: Dr. Pedro Cabalar Fernández
Director da Tese

To my parents

Logic is the beginning of wisdom, not the end.
Leonard Nimoy.

Logic, like whisky, loses its beneficial effect when taken in too large quantities.
Lord Dunsany.

Acknowledgments

Even though my name appears on the cover of this book, this thesis would not have been possible without the contributions of many people who deserve my infinite gratitude.

First of all, I would like to express my gratitude to my friend and supervisor Pedro Cabalar. Without his patience, direction, support and constant dedication this work would not have seen the light. You started TEL, you helped me to develop it and, moreover, we both shared great moments during these years. I owe you more than a thesis.

I would like to make a special mention to my colleagues Conchi, Feli and Gilberto. One guy started this thesis but it is another one who is writing it and this is because of your help, motivation and, of course, the lessons of life Gilberto taught me. I owe you more than a thesis, too.

I am also very thankful to those people with whom I shared the lab, laughter and memorable moments: Dani, Javi, Edu, Xose, Isma, Álvaro, Santos and Ana. Thank you for your support, advice and encouragement. I am also very grateful to Luis Fariñas, Philippe Balbiani, Andreas Herzig and, of course, my good friend Naji Obeid for their kindly hospitality and help during my stay at the IRIT in Toulouse.

From the academical point of view, this work would not exist without the contributions and suggestions of many researchers. Among others, David Peace, if he had not discovered Equilibrium Logic, this thesis would not make any sense. Stéphane Demri, the automata-based method he defined, led to important contributions in this thesis. Finally, Philippe Balbiani and Luis Fariñas, who showed me that, with the appropriate axioms, we can build great things.

Last but not least, I would like to say thanks to my parents and brother for their unconditional support (and patience) and also to my friends Roberto, Luis and Diego.

Abstract

Commonsense temporal reasoning is full of situations that require drawing default conclusions, since we rarely have all the information available. Unfortunately, most modal temporal logics cannot accommodate default reasoning, since they typically deal with a *monotonic* inference relation. On the other hand, non-monotonic approaches are very expensive and their treatment of time is not so well delimited and studied as in modal logic.

Temporal Equilibrium Logic (TEL) is the first non-monotonic temporal logic which fully covers the syntax of some standard modal temporal approach without requiring further constructions. TEL shares the syntax of *Linear-time Temporal Logic* (LTL) (first proposed by Arthur Prior and later extended by Hans Kamp) which has become one of the simplest, most used and best known temporal logics in Theoretical Computer Science.

Although TEL had been already defined, few results were known about its fundamental properties and nothing at all on potential computational methods that could be applied for practical purposes. This situation unfavourably contrasted with the huge body of knowledge available for LTL, both in well-known formal properties and in computing methods with practical implementations. In this thesis we have mostly filled this gap, following a research program that has systematically analysed different essential properties of TEL and, simultaneously, built computational tools for its practical application. As an overall, this thesis collects a corpus of results that constitutes a significant breakthrough in the knowledge about TEL.

Resumen

El razonamiento temporal del sentido común está lleno de situaciones que requieren suponer conclusiones por defecto, puesto que raramente contamos con toda la información disponible. Lamentablemente, la mayoría de lógicas modales temporales no permiten modelar este tipo de razonamiento por defecto debido a que, típicamente, se definen por medio de relaciones de inferencia monótonas. Por el contrario, las aproximaciones no monótonas existentes son típicamente muy costosas pero su manejo del tiempo no está tan bien delimitado como en lógica modal.

Temporal Equilibrium Logic (TEL) es la primera lógica temporal no monótona que cubre totalmente la sintaxis de alguna de las lógicas modales tradicionales sin requerir el uso de más construcciones. TEL comparte la sintaxis de *Linear-time Temporal Logic* (LTL) (formalismo propuesto por Arthur Prior y posteriormente extendido por Hans Kamp), que es una de las lógicas más simples, utilizadas y mejor conocidas en Teoría de la Computación.

Aunque TEL había sido definido, muy pocas propiedades eran conocidas, lo que contrastaba con el vasto conocimiento de LTL que está presente en el estado del arte. En esta tesis hemos estudiado diferentes aspectos de TEL, una novedosa combinación de lógica modal temporal y un formalismo no monótono. A grandes rasgos, esta tesis recoge un conjunto de resultados, tanto desde el punto de vista teórico como práctico, que constituye un gran avance en lo relativo al conocimiento sobre TEL.

Resumo

O razoamento do sentido común aplicado ao caso temporal está cheo de situacións que requiren supoñer conclusións por defecto, posto que raramente contamos con toda a información dispoñible. Lamentablemente a maioría de lóxicas modais temporáis non permiten modelar este tipo de razoamento por defecto debido a que, típicamente, están definidas por medio de relacións de inferencia monótonas. Pola contra, as aproximacións non monótonas existentes son moi costosos e o seu tratamento do tempo non está ben tan delimitado nin estudiado como nas lóxicas modais.

Temporal Equilibrium Logic (TEL) é a primeira aproximación non monótona que cubre totalmente a sintaxe dalgunha das lóxicas modais traidicionáis sen requerir o uso de máis construccións. TEL comparte a sintaxe de *Linear-time Temporal Logic* (LTL) (formalismo proposto por Arthur Prior e extendido posteriormente por Hans Kamp), que é considerada unha das lóxicas modais máis simples, utilizadas e coñecidas dentro da Teoría da Computación.

Aínda que TEL xa fora definido previamente, moi poucas das súas propiedades eran coñecidas, dato que contrasta co vasto coñecemento de LTL existente no estado da arte. Nesta tese, estudiamos diferentes aspectos de TEL, unha novidosa combinación de lóxica modal temporal e un formalimo non monótono. A grandes rasgos, esta tese recolle un conxunto de resultados, tanto dende o punto de vista teórico como práctico, que constitúe un gran avance no relativo ó coñecemento sobre o formalismo TEL.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Hans Kamp began his famous PhD thesis [63] on temporal logic with the following sentence:

> "Greece is a kingdom, is true now (May 10, 1968); but it has been false before and who knows if it will be true tomorrow?"

With this sentence, Kamp was emphasizing that the truth of a sentence may depend on the moment at which it is asserted. He was probably suspecting that King Constantine II of Greece, already reigning in the exile in 1968, would eventually cease, as it actually happened on June 1st, 1973 with the end of the Greek monarchy. If we were asked about the Greek regime at any time point between Kamp's thesis and June 1973 we would obviously conclude that it was a monarchy because we have no further evidence on the contrary. This is an example of default conclusion that can be retracted on the light of new facts. Consider now, for instance, the case of King Baudouin of Belgium, who officially reigned from July 17th, 1951 to July 31st 1993. Without further information we can assume that he was ruling at any date in that period. However, it is known that on April 4th, 1990, he abdicated[1] one day, in agreement with the Belgian Government, to avoid signing an abortion law. After adding this fact, we can retract the previous conclusion to derive that he was not ruling on that exceptional date.

Commonsense temporal reasoning is full of situations that require drawing default conclusions, since we rarely have all the information available. Unfortunately, neither Kamp's nor most modal temporal logics can accommodate default reasoning, since they typically deal with a *monotonic* inference relation (as happens in classical logic too). Formally, this means that if a formula φ is derived from a theory Γ, then φ will be still derivable from any theory of the form $\Gamma \cup \Delta$. In other words, a conclusion φ obtained from Γ *cannot be retracted* after adding new evidence Δ.

Computational logic has been applied to temporal reasoning under two different perspectives: *Theoretical Computer Science* (TCS) and *Knowledge Representation* (KR). Since the goals from these two areas are quite different, the results obtained in both cases have traditionally diverged. In TCS, temporal

[1] To be precise, he was declared "unable to reign" for one day.

reasoning is mostly focused on studying properties about algorithms and their related problems such as computability, complexity assessment, formal verification and connection to other frameworks such as automata theory, algebra or formal languages. In all these cases, the use of different types of *modal temporal logics* has proved to be extremely useful, since they provide specialised temporal constructions to talk about time while they usually restrict the expressiveness of full predicate calculus to achieve decidability. In KR, on the contrary, research has been traditionally guided by commonsense reasoning problems where defaults and *Non-Monotonic Reasoning* (NMR) played a crucial role. In particular, this was the main focus in the KR area of *Reasoning about Actions and Change* (RAC) where the most prominent formal languages are actually dialects of (many-sorted) first-order logic, since they seek a rich expressive capability.

While non-monotonicity has been practically out of the discourse in TCS, its combination with modal logics in KR is not rare. For instance, there exists a whole family of *modal non-monotonic logics* (see for instance [87]) that are obtained by imposing a fixpoint condition on the inference relation of standard monotonic modal frameworks. However, modalities in this context have mostly had an epistemic usage, representing concepts such as knowledge, belief or obligation and exclusively handling the standard operators of necessity and possibility. Combinations of NMR with modal logics specifically designed for temporal reasoning is much more infrequent in the literature. The few exceptions are typically modal action languages with a non-monotonic semantics defined under some syntactic limitations.

To the best of our knowledge, the only case of non-monotonic logic that fully covers the syntax of some standard modal temporal approach without requiring further constructions is *Temporal Equilibrium Logic* (TEL), introduced by Cabalar and Pérez in [23]. TEL shares the syntax of *Linear-time Temporal Logic* (LTL) (first proposed by Arthur Prior [103] and later extended by Hans Kamp [63]) which has become one of the simplest, most used and best known temporal logics in TCS. The main difference of TEL with respect to LTL lies in its non-monotonic entailment relation (obtained by a models selection criterion) and in its semantic interpretation of implication and negation, closer to intuitionistic logic. These two properties are actually inherited from the fact that TEL is a temporal extension of *Equilibrium Logic* [98], a non-monotonic formalism that actually provides a generalisation of the well-known *stable models* semantics [48] from logic programming to the case of arbitrary propositional formulas. This semantic choice is a valuable feature for different reasons. First, stable models constitute the basis of a successful NMR paradigm for practical problem solving, *Answer Set Programming* (ASP), with several efficient solvers available and many application domains. Second, unlike the original definition of stable models, the semantics of *Equilibrium Logic* does not depend on syntactic transformations but, on the contrary, is just a simple minimisation criterion for an intermediate logic (the logic of *Here-and-There* [57]). This purely logical definition provides an easier and more homogeneous way to extend the formalism, using standard techniques from other hybrid logical approaches. Thus, for instance, the temporal extension in TEL can practically be seen as the "obvious" way to introduce linear-time

temporal operators in Equilibrium Logic.

The first pair of works on TEL explored its use to translate action languages [23] or its potential application for checking (strong) equivalence between alternative representations of some action domain [2]. However, apart from TEL definition itself, few results were known about its fundamental properties [17] and nothing at all on potential computational methods that could be applied for practical purposes. This situation unfavourably contrasted with the huge body of knowledge available for LTL, both in well-known formal properties and in computing methods with practical implementations.

In this thesis we have mostly filled this gap, following a research program that has systematically analysed different essential properties of TEL and, simultaneously, building computational tools for its practical application. In this way, we have provided different translations of TEL into other formalisms and formally compared to other related approaches, we have partially axiomatised its monotonic basis (the logic of *Temporal Here-and-There*), we have proved that the latter is an adequate monotonic basis (by showing a result on so-called *strong equivalence*) and we have proved that the set of TEL models of a theory can be captured by standard models of some LTL formula. On the other hand, all these fundamental results have also allowed us to develop a pair of computational tools: one, called `STeLP`, that allows obtaining temporal equilibrium models of a syntactic subset of TEL (a kind of temporal logic programs allowing variables), and a second one, called `ABSTEM`, that is applicable to any arbitrary (propositional) syntax and further allows checking different types of equivalence between a pair of theories. In both cases, the tools combine different backends coming from LTL, automata theory and ASP.

One of the main difficulties of this work has been dealing with concepts and theories coming, on the one hand, from LTL and automata theory in TCS, and on the other hand, from ASP logical encodings and NMR in KR. For a better understanding of the context, let us provide a brief overview on the general related literature in these areas.

1.1 Approaches for temporal reasoning

1.1.1 Modal Temporal Logics

The first definition of a modal logic for dealing with time is Arthur Prior's [103] *Tense Logic*, that introduced two new modal operators for respectively referring to past and future events. In Prior's system, time can be mathematically represented as a structure $\langle T, < \rangle$ where T is a set of time points and $<$ a binary relation among them that can be linear and dense, but not branching. Kamp [63] incorporated to Tense Logic two new temporal operators, *until* and *since*, and was able to prove that this extended system, we call nowadays *Linear Time Temporal Logic* (LTL), is *expressively complete* with respect to a fragment of First Order Logic called *Monadic First Order Logic of Linear Order*, or $MFO(<)$ for short. Later on, several alternative proofs of this result have been published, using different approaches such as separation property [43, 42], game theory [58] and more recently by means of a syntactic subclass of first-order formulas [104].

The application of LTL in Computer Science was first proposed by Amir Pnueli [102], who exploited this formalism as a suitable framework for reasoning about reactive systems. After that, temporal logic became more popular in Computer Science causing an extensive research for the last 30 years. Several approaches oriented to program and protocol verification were proposed, being perhaps the most popular frameworks the already mentioned LTL, *Computational Tree Logic* (CTL) [28] and *Propositional Dynamic Logic* (PDL) [39]. While LTL deals with linear time, CTL represents the future as a tree with different alternative paths. These two logics have differences in their expressive power (we cannot capture one on the other) but they are both fragments of CTL* [33] which can be in its turn embedded into the more general μ-calculus [67] dealing with fixpoint constructs. On the other hand, the main feature of PDL is that its modal connectives allow representing not only different elementary actions, but also any arbitrary program containing sequential, alternative and iterative constructs. Different extensions of PDL have been studied and axiomatised [114, 9], whereas it is known that this logic can be also encoded in μ-calculus.

Modal temporal logics have the important advantage of providing, in most cases, associated computation methods that are decidable. Furthermore, these methods usually rely on automata construction or manipulation of formal languages, showing in this way a clear connection to other well-known mathematical formalisms. The choice of LTL for the purpose of this thesis has some clear advantages. The first of them is its simplicity, as it deals with a simple structure for time points (a linear order) while it provides a small set of modal connectives with a quite natural reading. Another advantage is that it is one of the oldest modal temporal approaches, with a large body of knowledge on fundamental properties and computation methods, and being in many cases the first modal logic on which some well-established techniques in the area have actually been tried out. For instance, its expressiveness is well delimited: as said before, it is known to be as expressive as the fragment $MFO(<)$ of First Order Logic, but strictly less expressive than the second order version of this fragment, called $MSO(<)$. Its axiomatisation was tackled in [121, 53], its satisfiability is known to be a PSPACE-complete problem [111], and there exist well-known automata construction methods [120] that have allowed nowadays its extensive use for model checking [59] and verification of reactive systems such as software, communication protocols or hardware circuits. Besides, it has been successfully used or combined with formalisms from KR and Artificial Intelligence such as languages for Multi-Agent Systems [40], Ontologies and Description Logics [6, 86] or as a language for constraint specification in planning systems [7, 69, 8]. However, in none of these cases, LTL was actually combined with NMR, even though the latter has played a central role in KR research, as we explain next.

1.1.2 Reasoning about Actions and Change

One of the main goals of Artificial Intelligence (AI) is providing computer programs that perform high-level reasoning tasks such as planning, explanation or diagnosis. Quoting John McCarthy, while humans may learn these abilities,

> "In order for a program to be capable of learning something it must first be capable of being told it." [89].

This means that an artificial "mind" should be equipped with an abstract representation of the different entities involved in its environment and the commonsense rules that govern them. The AI area that studies this problem receives the name of Knowledge Representation (KR). Following [113], KR constitutes a "multidisciplinary subject that applies theories and techniques from three other fields: (1) Logic provides the formal structure and rules of inference; (2) Ontology defines the kinds of things that exist in the application domain; (3) Computation supports the applications that distinguish knowledge representation from pure philosophy." The use of Logic as a predominant tool in KR, is not surprising since it fulfills many of the requirements of the field that other languages satisfy to a less degree or in some cases simply do not accomplish: accuracy, lack of ambiguity, clear semantics, simplicity and readability, coverage/expressiveness, flexibility, together with inference methods and their associated algorithms.

The early attempts of formalisations of commonsense reasoning soon unveiled that Classical Logic had an important limitation in one of the criteria previously mentioned: flexibility or, using a more specific term, what John McCarthy called *elaboration tolerance*. In McCarthy's words:

> "A formalism is *elaboration tolerant* to the extent that it is convenient to modify a set of facts expressed in the formalism to take into account new phenomena or changed circumstances." [92]

Historically, the KR area of Reasoning about Actions and Change (RAC) has gradually analysed different representational problems related to a lack of elaboration tolerance, being the so-called *frame problem* [93] the first one in being detected and probably one of the most influential ones. The frame problem consists in the difficulty of explicitly representing all the facts that should remain unchanged after the execution of an action. Its solution requires some form of NMR that allows encoding the so-called default rule of *inertia*: things remain unchanged along time unless there is evidence on the contrary. During RAC evolution, other representational issues have been studied and solved using NMR such as the *qualification problem* (that has to do with exceptions to actions executability) or the *ramification problem* (related to the treatment of indirect effects).

The central role of NMR in the solutions to these representational problems from the area of RAC boosted the interest on the definition and properties of different NMR techniques. In 1980, a special issue on the topic of the *Artificial Intelligence* journal introduced three well-known NMR formalisms that were extensively studied afterwards: *Circumscription* [90], *Default Logic* [105] and *Non-monotonic modal logics* [106]. These NMR approaches were then used in the semantics of action languages. In particular, for instance, different forms of predicate Circumscription were applied in *Situation Calculus* [93], *Event Calculus* [66] and *Features and Fluents* [108]. It is perhaps interesting to note that most approaches in RAC have traditionally represented time as one more sort inside a many-sorted First Order Logic formalization. For instance, in the case of Situation Calculus, we find a sort for *situations*, formally represented

by syntactic terms of the form $do(a, s)$ where a is some action and s, in its turn, a situation term or the constant S_0 (the initial situation). In order to represent a time-dependent predicate $P(x)$ (called *fluent* in RAC terminology) it becomes reified as an argument of a temporal "metapredicate" $Holds(P(x), s)$ where s is a situation term. In this way, we get a very expressive formalism without limitations on the quantification of situations or the construction of arbitrary expressions involving them. However, in the general case, we inherit the undecidability of First Order Logic and a great part of modern research related to Situation Calculus has actually involved the delimitation of different kinds of decidable subclasses or reasoning problems for that formalism (see, for instance, [50]). The case of Event Calculus replaces the use of situations by a sort of *events* and a richer family of temporal metapredicates that can assert properties about the truth of fluents ($Initiaties$, $Terminates$, etc) but is formally similar in the use of First Order Logic.

This predominance of First Order Logic in RAC is not surprising, given the general goals of KR and its search for expressive languages. On the contrary, the use of modal logics[2] for representing time in such an area has been much more unfrequent. As a pair of exceptions (we will comment in Chapter 7) we can mention [24] and [51], based on variants of Dynamic Logic.

1.1.3 Answer Set Programming

An important breakthrough in the area of NMR was the discovery of a strong connection [12] between Default Logic and Logic Programming (LP) that allowed reading the Prolog operator of negation-as-failure as an NMR formalisation of default negation. This opened the possibility of reinterpreting logic programs as KR formalisms with embodied NMR, leading to a cross fertilization between both areas. Among the different LP semantics proposed for negation-as-failure, one of them has eventually had a striking impact for practical application of NMR. In [48], Michael Gelfond and Vladimir Lifschitz introduced the so-called *stable model* semantics that subsumed many of the existing LP alternatives but, very importantly, without the syntactic restrictions made by previous approaches. The model-based orientation of this semantics eventually led to a paradigm suitable for constraint-satisfaction problems that is known nowadays as *Answer Set Programming* (ASP) [96, 88] and that became one of the most prominent and successful approaches for practical KR. Great part of this success is due to the impressive advances in implementation of efficient solvers [73, 45] and development of applications. In this aspect, ASP shows some similarities to SAT (propositional satisfiability), since it is also targeted for solving problems in the NP complexity class (at least when we restrict to non-disjunctive ASP) and there already exists a regular ASP solver competition [5]. However, unlike SAT, a crucial factor in ASP which is also responsible for its success is its versatility for knowledge representation. While most applications of SAT involve an external translation of the problem to solve into clauses in propositional logic, ASP provides a powerful language based on lo-

[2] John McCarthy, the founder of logical KR and research on formal commonsense reasoning, showed in several occasions an explicit disapproval of modal logics. See for instance his position paper with the self-explanatory title "Modality, si! Modal logic, no!" [91].

gic program rules with variables, allowing non-monotonicity through default negation and offering additional expressive features such as disjunction in the head, aggregates, constraints on numeric variables, functions, etc. As a pair of ASP applications especially remarkable from the KR perspective we can mention the decision support system for the Space Shuttle [97] and the information integration project INFOMIX (see [14] for an updated overview of the area and more application examples).

But the success of ASP does not just amount to its practical applications: during its evolution many hints have pointed out its relevance inside the theoretical foundations of NMR. For instance, in [76] Lifschitz enumerates up to 12 known alternative definitions of a stable model (and there exist more) coming from different areas such as NMR (including translations into Default Logic, Autoepistemic Logic and Circumscription), logic programming and non-classical logics. Inside the last category, one approach has had a particular success in the study of foundations of ASP: the logical characterisation based on Equilibrium Logic introduced by David Pearce [99]. This characterisation has shown interesting features. For instance, it has naturally covered the syntactic extensions that had been proposed for stable models and, going further, allowed defining stable models for any arbitrary theory in the syntax of First Order Logic [38, 101]. Second, it has been shown to be crucial [78] in fully capturing the important property of strong equivalence, an NMR property that means that two theories not only yield the same models, but also will keep doing so when new information is added. Third, as Equilibrium Logic provides a purely logical definition, it has allowed the application of logical techniques and results to ASP including the possibility of defining hybrid logical approaches (like the one we study in this thesis).

Despite of all these features, there exists one important aspect in knowledge representation that had not been included among the usual ASP language extensions: logical constructs for *temporal reasoning*. Temporal scenarios are very frequent in ASP applications and benchmarks. The use of ASP for action domains was already proposed by Gelfond and Lifschitz in [49], which established a methodology later followed by other action languages defined as front-ends for ASP [32, 47, 44]. In fact, ASP offers essential features for a suitable formal representation of temporal scenarios. For instance, it allows a simple and natural solution to the frame, ramification and qualification problems by a suitable use of logic programming rules plus default negation. Another interesting feature is that it allows a uniform treatment of different kinds of reasoning problems such as prediction, postdiction, planning, diagnosis or verification. However, since ASP is not a temporal formalism, it also involves some difficulties for dealing with temporal problems. The methodology mostly adopted in ASP has been inherited from that first proposed in [49] based on transition systems. Time is represented as an extra argument (normally, an integer variable[3]) for predicates whose truth may vary in each transition. A simple example of a dynamic scenario in ASP can be found in Section 2.1.7. Since most ASP tools must deal with *finite domains*,

[3]As an exception, it is worth perhaps to mention the implementation of Situation Calculus presented in [72] that uses the first order ASP tool F2LP [71].

this requires fixing a finite plan length[4] with an obvious impossibility for solving problems such as proving the non-existence of a plan for a given planning scenario or verifying temporal properties of the transition system behaviour.

From the previous discussion, it seems that a reasonable choice would be keeping all the nice knowledge representation features from ASP while taking benefit from the advances in the area of modal temporal logic. There exists a pair of approaches that combine ASP with temporal logic: the already mentioned Temporal Equilibrium Logic (TEL), that uses LTL as modal approach and is the main focus of this thesis, and *Temporal Answer Sets* (TAS) [51], that uses *Dynamic Linear Temporal Logic* (DLTL) [51] instead. An important difference between these two approaches is that while the latter's semantics is defined in terms of a syntactic transformation on a restricted syntactic class of formulas, the former provides a non-monotonic semantics exclusively based on a model selection criterion that covers any arbitrary theory in the syntax of LTL. A deeper discussion on the relation between TEL and TAS is included in Section 7.2, which even includes a formal comparison. We provide now the main goals of this work and explain the structure of this document.

1.2 Goals and structure of this thesis

In this thesis we will develop several important aspects of the TEL formalism that were previously unknown or unexplored, covering both fundamental properties and implementation topics. The goals of this work can be enumerated as follows:

- Fixing relevant *theoretical properties* of TEL. We will study, among others, the following topics:
  - We will provide different translations of TEL into other logics, such as regular LTL, *Quantified Equilibrium Logic* [101] or *Infinitary Equilibrium Logic* [55].
  - We will axiomatise (a relevant fragment) of the monotonic basis of TEL (the logic of *Temporal Here-and-There*, THT).
  - We will prove the adequacy of THT as a monotonic basis by proving that the equivalence in this logic is a necessary and sufficient condition for proving the *strong equivalence* of two theories in TEL, i.e., that the two theories are interchangeable regardless of the context.
- Defining and implementing procedures that allow *computing temporal equilibrium models* as well as checking several desired properties of a given temporal specification. In particular, we study the construction of tools for:
  - obtaining temporal equilibrium models of a syntactic subset of TEL (a kind of temporal logic programs allowing variables);
  - studying grounding algorithms for removing variables in temporal programs;

[4] In a similar technique to planning as propositional satisfiability [65].

  - obtaining temporal equilibrium models for arbitrary theories in the syntax of (propositional) LTL;
  - checking different types of equivalence between a pair of theories and showing information (countermodels) when the theories are not equivalent.

- Compare TEL with other non-monotonic approaches for representing temporal scenarios, including formalisms from temporal logic programming, modal approaches from RAC and the related temporal ASP approach *Temporal Answer Sets* [51], for which we tackle a formal comparison.

The general methodology of this proposal will be the standard in research in Computer Science, a cyclic sequence including: review of the state-of-the-art, problem definition, posing hypotheses, deriving their formal proof or rebuttal, testing experiments on benchmarks (in case of implementation topics) and finally evaluation and publication. Many results have been accompanied by software prototypes to test their behavior and practical feasibility. For the development of these prototypes (especially in the cases of `STeLP` and `ABSTEM`), we have applied a spiral life-cycle passing through different cycles of analysis, design, implementation and evaluation. Regarding to publication, this thesis has yielded the following published results:

- P. Cabalar and M. Diéguez. *STeLP - A Tool for Temporal Answer Set Programming*, in Proc. of the 11th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'11), pages 370–375, Vancouver, Canada, 2011.

- P. Cabalar and M. Diéguez. *Strong equivalence of non-monotonic temporal theories*, in Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14), Vienna, Austria, 2014.

- F. Aguado, P. Cabalar, M. Diéguez, G. Pérez, C. Vidal, *Temporal equilibrium logic: a survey*. Journal of Applied Non-Classical Logics 23(1-2): 2-24 (2013).

- F. Aguado, P. Cabalar, M. Diéguez, G. Pérez and C. Vidal, *Paving the Way for Temporal Grounding*, in Proc. of the 28th International Conference on Logic Programming (ICLP'12), Budapest, Hungary, 2012.

The structure of this thesis will be the following: in Chapter 2 we introduce some background about Answer Set Programming, Equilibrium Logic and Linear Temporal Logic as well as several well-known results that will be used along this thesis. In Chapter 3 we present Temporal Equilibrium Logic, some known properties and its connection with both Quantified Equilibrium Logic and *Infinitary Formulas*. In Chapter 4 we set the bases for the axiomatisation of Temporal Here and There by adapting several classical methods to prove completeness of classical modal logics to deal with this kind of hybrid approaches. In Chapter 5 we present two tools, `STeLP` and `ABSTEM` for computing temporal

equilibrium models of temporal theories. In Chapter 6 we prove that THT is a suitable monotonic base to check strong equivalence, which leads to extend the functionalities of `ABSTEM` with the possibility of checking several kinds of temporal equivalence. Finally, in Chapter 7, we compare TEL to other approaches of nonmonotonic reasoning that are present in the literature and we finish this thesis with some conclusions and future work in Chapter 8.

Chapter 2

Background

2.1 Answer Set Programming

In [12] an important connection was discovered: if we momentarily forget the control strategy of Prolog, we may essentially understand that a program with negation as failure like, let us say:

```
p :- not  q.
r :- p, not  s.
```

tries to solve the same problem as the theory in Default Logic

$$\frac{: \neg q}{p} \qquad \frac{p : \neg s}{r}$$

This relation opened the cross research between the fields of nonmonotonic reasoning and logic programming. On the one hand, nonmonotonic approaches are used to study declarative semantics for logic programming and its generalizations. On the other hand, logic programs provide an excellent tool for obtaining practical applications of nonmonotonic reasoning.

Answer Set Programming [14] (ASP), probably the most successful semantics for logic programming, has emerged as a powerful paradigm for knowledge representation and reasoning. Based on Non-monotonic Reasoning and Stable Models Semantics [48] this language has become a suitable framework for modelling knowledge intensive applications like for example, diagnosis, commonsense reasoning or temporal reasoning. In this section we study this paradigm beginning with a description of *normal ASP logic programs*.

2.1.1 ASP logic programs

A *normal logic program* consists of a set of implications of the form

$$a \leftarrow b_1, \cdots, b_m, not\ b_{m+1}, \cdots, not\ b_n \tag{2.1}$$

where a is either an atom or $\bot$ and and each b_i is an atom from some signature At. If $a = \bot$ the rule is called *constraint* while if both $m = 0$ and $n = 0$, the

rule can be seen as the implication

$$a \leftarrow \top$$

and it is called *fact*. Both $\top$ and $\bot$ are the constants *true* and *false* of classical logic. Usually, the antecedent of the implication is known as *body* whereas the consequent is called *head*. We will often represent a rule r of the form

$$head(r) \leftarrow body^+(r), body^-(r)$$

where $head(r) = \{a\}$, $body^+(r) = \{b_1 \wedge \cdots \wedge b_m\}$ and $body^-(r) = \{\neg b_{m+1} \wedge \cdots \wedge \neg b_n\}$, obtained from replacing the operator ',' by $\wedge$ and default negation, '*not*', by $\neg$. We usually identify $head(r)$, $body^+(r)$ and $body^-(r)$ with their corresponding sets of atoms. Finally, the set of atoms that occur in a rule r is denoted by $atoms(r)$.

A normal logic program is said to be *positive* (or *definite*) when it does not contain any default literal. Note that, if we interpret ',' and '$\leftarrow$' as classical conjunction and implication, then a positive logic program just corresponds to a set of Horn clauses. As an example of positive program, we present the following program, which will be analysed in greater detail during this section.

Example 2.1 (from [14])**.**

$$high\_salary \leftarrow employed, educated. \quad (2.2)$$
$$educated \leftarrow high\_salary. \quad (2.3)$$
$$employed \leftarrow motivated. \quad (2.4)$$
$$motivated. \quad (2.5)$$

⊠

Regarding this program as a set of rules in logic programming, it would mean that $motivation$ is a precondition to be $employed$ as well as $high\_salary$ is necessary to derive $educated$. Moreover, both $employed$ and $educated$ are needed to derive $hight\_salary$.

Definition 2.1 (model)**.** *We say that an interpretation I satisfies a positive rule*

$$c \leftarrow a_1, \cdots, a_n$$

iff when $a_i \in I$ for $i = 1 \cdots n$. then $c \in I$ too. We say that I is a model of a positive program Π*, written $I \models \Pi$, iff I satisfies all the rules in* Π. ⊠

2.1.2 Stable model semantics

The definition of *stable models* [48] semantics naturally arises from the already explained connection between default logic and logic programming. For a positive program Kowalski and van Emden [117] proved that there exists a unique minimal model called *Least Herbrand Model* and, furthermore, they showed that it can be computed by means of T_Π operator, defined below, starting with $I = \emptyset$ and applying it iteratively until reaching a fixpoint:

$$T_\Pi(I) = \{c \mid (c \leftarrow a_1, \cdots, a_n) \in \Pi \text{ and } a_i \in I \text{ for all } i \in [1, n]\}$$

Proposition 2.1 (from [117])**.** *Given a positive logic program* Π*, the least fixpoint of* T_Π *is the* least model *of* Π *denoted by* $LM(\Pi)$*.* ⊠

For instance, the minimal model of program (2.2)-(2.5) corresponds to the set $\{motivated,\ employed\}$.
However, when we introduce default negation we cannot guarantee the existence of a unique minimal model and, moreover, the techniques described before are not directly applicable. For instance let us consider the program (2.2)-(2.5) plus the following pair of rules with default negation:

$$educated \leftarrow not\ illiterate. \tag{2.6}$$
$$illiterate \leftarrow not\ educated. \tag{2.7}$$

The information we had derived before adding these rules is still valid, so we can still determine that both $motivated$ and $employed$ have to be true. However, we cannot conclude anything about the literals $educated$ and $illiterate$ since we do not know which atoms cannot be derived and, therefore, we cannot verify the conditions for applying any of the rules. A way out of the problem is to start by assuming which atoms will not be derived. For instance, let us assume that $illiterate$ will not be derived. Then, the first rule can be used and we can establish $educated$ and, as a consequence, $high\_salary$ as true. Once $educated$ is stablished, the rule (2.7) cannot be used and $illiterate$ will not be stablished, verifying our assumption. On the other hand, if we assume that $educated$ cannot be derived, then rule (2.7) can be used to derive $illiterate$, which makes that rule (2.6) cannot be used to derive $illiterate$, justifying our assumption. It should be noticed that the assumption of both $illiterate$ and $educated$ will not be derived cannot be justified by the program, as done in the previous cases. We provide next a formal definition that captures this idea.

Definition 2.2 (program reduct from [48])**.** *Given a set of atoms* I *and a logic program* Π*, the* program reduct *with respect to* I*, denoted by* Π^I*, is a set of positive rules of the form*

$$head(r) \leftarrow body^+(r)$$

such that there is a rule

$$head(r) \leftarrow body^+(r), body^-(r)$$

in Π *satisfying* $body^-(r) \cap I = \emptyset$*.* ⊠

As an example, consider the program consisting of rules (2.2)-(2.7) and the interpretation $I = \{educated\}$. Π^I would correspond to the following positive program:

$$high\_salary \leftarrow employed, educated. \tag{2.8}$$
$$educated \leftarrow high\_salary. \tag{2.9}$$
$$employed \leftarrow motivated. \tag{2.10}$$
$$motivated. \tag{2.11}$$
$$educated. \tag{2.12}$$

Note that, since Π^I is positive, it has a unique model that corresponds to the Least Model, denoted by $LM(\Pi^I)$, and can be computed by means of T_Π operator.

Definition 2.3 (stable model)**.** *Let I be an interpretation for a normal logic program (possibly with negation)* Π*. I is a stable model of* Π *if* $I = LM(\Pi^I)$*.* ⊠

By applying this definition it easy to check that the stable models of the program (2.2)-(2.7) are

$$\{motivated, employed, educated, high\_salary\}$$

and

$$\{motivated, employed, illiterate\}.$$

2.1.3 Splitting a logic program

When we gave the informal explanation of how to compute the stable models of the program (2.2)-(2.7) we said that the true atoms concluded from the evaluation of (2.2)-(2.5) were still valid. By this first line of reasoning we somehow isolated a subprogram that was evaluated and used afterwards in order to simplify the rest of the rules. This process is formally known as *splitting* [82] and it can be used to simplify the computation of the stable models of a program.

Sometimes, a logic program Π can be divided into two parts, called *top* and *bottom*, such that there is no reference from the bottom part to the top one.

Definition 2.4 (splitting set from [82])**.** *A splitting set for a program* Π *is any set U of atoms such that, for every rule $r \in \Pi$, if $head(r) \cap U \neq \emptyset$ then $atoms(r) \subset U$. If U is a splitting set for* Π *we also say that U splits* Π*. The set of rules $r \in \Pi$ such that $lit(r) \subset U$ is called the* bottom *of* Π *relative to the splitting set U and denoted by $b_U(\Pi)$. Consequently, the set $\Pi \setminus b_U(\Pi)$ is the* top *of* Π *relative to U.* ⊠

To give an example, let us consider the program below:

$$c \leftarrow a \tag{2.13}$$
$$d \leftarrow b \tag{2.14}$$
$$a \leftarrow not\ b \tag{2.15}$$
$$b \leftarrow not\ a \tag{2.16}$$

The set $U = \{a, b\}$ is a splitting set for Π being $b_U(\Pi) = \{(2.15), (2.16)\}$. The idea of splitting is that we can compute first each stable model X of $b_U(\Pi)$ and then use the truth values in X for simplifying the program $\Pi \setminus b_U(\Pi)$ from which the rest of truth values for atoms not in U can be obtained. Formally, given $X \subseteq U \subseteq At$ and an ASP program Π, for each rule r like (2.1) in Π such that $(body^+(r) \cap U) \subseteq X$ and $body^-(r) \cap U$ is disjoint from X, take the rule

$$r^\bullet : head(r) \leftarrow body^+(r)^\bullet \wedge body^-(r)^\bullet$$

where $body^+(r)^\bullet = (body^+(r) \setminus U)$ and $body^-(r)^\bullet = (body^-(r) \setminus U)$. The program consisting of all rules $r^\bullet$ obtained in this way is denoted as $e_U(\Pi, X)$. Note that this program is equivalent to replace, in all rules of Π, each atom $p \in U$ by $\bot$ if $p \notin X$ and by $\top$ if $p \in X$.

In the previous example, the stable models of $b_U(\Pi)$ are $\{a\}$ and $\{b\}$. For the first stable model $X = \{a\}$, we get $e_U(\Pi \setminus b_U(\Pi), \{a\}) = \{c \leftarrow \top\}$ so that $X \cup \{c\} = \{a, c\}$ should be a stable model for the complete program Π. Similarly, for $X = \{b\}$ we get $e_U(\Pi \setminus b_U(\Pi), \{b\}) = \{d \leftarrow \top\}$ and a "completed" stable model $X \cup \{d\} = \{b, d\}$. The following result guarantees the correctness of this method in the general case.

Theorem 2.1 (from [82])**.** *Let U be a splitting set for a program Π consisting of a set of rules like* (2.1)*. A set of atoms X is a stable model of Π if, and only if both conditions hold:*

(i) *$X \cap U$ is a stable model of $b_U(\Pi)$,*

(ii) *$X \setminus U$ is a stable model of $e_U(\Pi \setminus b_U(\Pi), X \cap U)$.* ⊠

In [82], this result was generalised for an infinite sequence of splitting sets, showing an example of a logic program with variables and a function symbol, so that the ground program was infinite.

2.1.4 Completion

Apart from the Stable Models semantics, several other logic programming semantics have been proposed. For normal logic programs, Clark [27] introduced the concept of *completion*, which consists in capturing the behaviour of default negation by a translation into classical logic. In order to show the intuitive idea about completion let us consider the following normal logic program:

$$\begin{array}{lll} p & \leftarrow & q, s. \\ p & \leftarrow & t. \\ r. & & \end{array}$$

Althought from the point of view of classical logic, the truth value of p is free, that is it can be true or false depending on the model we consider, in logic programming the truth value of p can be only derived by applying the rules provided. Therefeore p will be true only iff the expression $(q \wedge s) \vee t$ holds. By extending this intuitive idea to all of the logic program, we can define its *completion* as the following classical theory:

Definition 2.5 (completion from [27])**.** *Let At be a set of atoms and Π a normal logic program built from atoms of At. The operator $COMP[\Pi]$ is formally defined as:*

$$COMP[\Pi] \stackrel{\text{def}}{=} \{p \leftrightarrow B_1 \vee \cdots \vee B_n \mid p \in At \text{ and } B_i \text{ are all the bodies for head } p\}.$$

Some conventions must be considered

1. *The empty body is considered as the constant* $\top$
2. *An empty disjunction corresponds to* $\bot$
3. *Logic programming operators such as "*not*" and "," are replaced by their classical correspondences "*$\neg$*" and* $\wedge$ *respectively.*

$\boxtimes$

Following the rules above, the previous example can be translated into the following classical theory:

$$p \leftrightarrow (q \wedge s) \vee t. \tag{2.17}$$
$$q \leftrightarrow \bot. \tag{2.18}$$
$$r \leftrightarrow \top. \tag{2.19}$$
$$s \leftrightarrow \bot. \tag{2.20}$$
$$t \leftrightarrow \bot. \tag{2.21}$$

whose only classical model is $\{r\}$ (which, in fact, coincides with its unique stable model). However, for the program $p \leftarrow p$, whose stable model is $\emptyset$, its corresponding completion theory would be $p \Leftrightarrow p$, which has two models: $\emptyset$ and $\{p\}$. From this example we can conclude that the completion of a logic program can lead to different models. For instance, let us consider the program, with default negation, of the Figure 2.1(a) and its completion 2.1(b). Although the program 2.1(a) has as stable model $\{a\}$, its completion is satisfied by $\{a, c, d\}$ and $\{a\}$. Models of completion are known as *supported models* and correspond to all fixpoints of the operator T_Π. It is easy to prove that any stable model of a normal logic program is also a supported model, but the opposite (as we have seen) does not hold.

$$\begin{aligned} a &\leftarrow not\ b, not\ c. \\ a &\leftarrow d. \\ c &\leftarrow d. \\ d &\leftarrow c. \end{aligned}$$

(a) Logic program Π.

$$\begin{aligned} a &\leftrightarrow (\neg b \wedge \neg c) \vee d \\ b &\leftrightarrow \bot \\ c &\leftrightarrow d \\ d &\leftrightarrow c \end{aligned}$$

(b) completion of Π.

Figure 2.1: A logic program Π and its completion.

The reader might wonder about a more accurate relation between stable models of a logic program and its completion. The answer was given by Fages [35], who showed that if a logic program satisfies certain syntactic condition its stable models coincide with the models of its completion. Such condition, called tightness, was generalised in [34].

2.1.5 Disjunctive logic programs

A *disjunctive logic program* is a finite set of (disjunctive) rules of the form

$$a_1 \vee \cdots \vee a_k \leftarrow a_{k+1}, \cdots, a_m, not\ a_{m+1}, \cdots, not\ a_n$$

where $n \geq m \geq k \geq 0$ and $a_1, \cdots, a_n$ are atoms. If $k = 0$, then this rule is called a *constraint*. As before, if $k \neq 0$, it is a *proper* rule, and if $k = 1$, it is a *normal* rule. A disjunctive rule can also be written as

$$head(r) \leftarrow body^+(r), body^-(r).$$

where $head(r)$ stands for the disjunction $a_1 \vee \cdots \vee a_k$ and both $body^+(r)$ and $body^-(r)$ as before. The definition of reduct for normal programs can be extended to the disjunctive case in an straightforward way:

Definition 2.6 (disjunctive reduct from [81])**.** *The reduct of a formula, rule or program relative to an interpretation X is defined recursively, as follows:*

- $F^X = F$ *for a literal* F.
- $(F \wedge G)^X = F^X \wedge G^X$
- $(F \vee G)^X = F^X \vee G^X$
- $(not\ F)^X = \begin{cases} \bot & if \quad X \models F^X \\ \top & otherwise \end{cases}$
- $(F \leftarrow G)^X = F^X \wedge G^X$
- $\Pi^X = \{(F \leftarrow G)^X \mid F \leftarrow G \in \Pi\}$.

⊠

Lemma 2.1 (from [81])**.** *An interpretation X is an answer set of a disjunctive program Π if X is an answer set of the reduct Π^X* ⊠

2.1.6 Loop formulas for disjunctive logic programs

As we pointed out in Section 2.1.4, the operator $COMP$ captures the semantics of default negation as a propositional formula but, as we showed in such section, it is not always applicable to obtain the stable models. However, Lin and Zhao [84] showed how to turn a logic program Π into a propositional formula Γ such that its classical models coincide with the answer sets of Π. To achieve this result, known as the Lin-Zhao theorem, they define the concept of "loop formulas", an specific set of propositional formulas such that, in conjunction with $COMP[\Pi]$, they capture exactly the answer sets of Π as a propositional formula. In [37] the Lin-Zhao theorem was simplified, avoiding the step of computing $COMP[\Pi]$ and, furthermore, generalised to the case of the disjunctive logic programs as well as the more general definition of stable models from [36]. Since the definition in [37] is more general we will follow this work, which starts by defining the concepts of dependency graph and loop respectively.

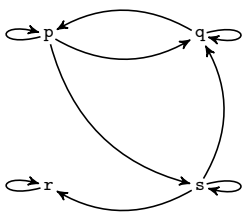
Definition 2.7 (loop from [37])**.** *Given a disjunctive logic program* Π*, its (positive) dependency graph* $G(\Pi)$ *is the directed graph whose vertices are the atoms occurring in* Π*. Every edge* (a_i, a_j) *in the graph represents a positive dependency in a rule* r *of* Π *if* a_i *occurs in its positive body and* a_j *in its head. Given the dependency graph of a program* Π*, denoted by* $G(\Pi)$*, a set of literals* L *is a called* loop *iff the subgraph of* $G(\Pi)$ *restricted to* L *is* Strongly Connected*.*

⊠

The main difference of the approach in [37] with respect to [84] is the consideration of *unary loops* (we assume that any atom is always connected to itself) in order to avoid computing the completion of Π. As an example, let us consider the program (from [37]) and its (positive) dependency graph shown in Figure 2.2.

$$
\begin{array}{rcl}
p & \leftarrow & q. \\
q & \leftarrow & p. \\
p & \leftarrow & \neg r. \\
r & \leftarrow & \neg p. \\
r \vee q & \leftarrow & s. \\
s & \leftarrow & p.
\end{array}
$$

(a) Π



(b) $G(\Pi)$

Figure 2.2: A disjunctive logic program Π and its corresponding $G(\Pi)$.

It is easy to see that the loops of Π are $\{p\}$, $\{q\}$, $\{r\}$, $\{s\}$, $\{p, q\}$ and $\{p, q, s\}$. Each loop will lead to a loop formula, which is an implication whose antecedent is the disjunction of the atoms in the loop and the consequent is called *external support* and defined next:

Definition 2.8 (external support from [37])**.** *Let* Π *be a disjunctive logic program and* L *a loop in* $G(\Pi)$*. The external support of* L*,* $ES_\Pi(L)$*, is the expression*

$$
ES_\Pi(L) \stackrel{\text{def}}{=} \bigvee_{r \in \Pi \text{ and } head(r) \cap L \neq \emptyset} \left(\bigwedge_{p \in body^+(r)} p \wedge \bigwedge_{q \in body^-(r)} q \wedge \bigwedge_{a \in (head(r) \setminus L)} \neg a \right)
$$

⊠

By following Definition 2.8, we can compute the set of loop formulas of Π (shown in Table 2.1) and obtain a final propositional theory consisting of $\Pi \cup LF(\Pi)$.

Loop	Loop Formula
$\{p\}$	$p \rightarrow q \vee \neg r$
$\{q\}$	$q \rightarrow p \vee (\neg r \wedge s)$
$\{r\}$	$r \rightarrow \neg p$
$\{s\}$	$s \rightarrow p$
$\{p, q\}$	$p \wedge q \rightarrow q \vee p \vee \neg r \vee (s \wedge \neg r)$
$\{p, q, s\}$	$p \wedge q \wedge s \rightarrow q \vee p \vee \neg r \vee (s \wedge \neg r) \vee p$

Table 2.1: Loop formulas obtained from program Π' of the Figure 2.2

Theorem 2.2 (from [37])**.** *Given a program* Π *for a signature* At *and a (classical) model* $X \subseteq At$ *of* Π *then* X *is a stable model of* Π *iff for every loop* L *of* $G(\Pi)$*,* X *satisfies* $\bigvee_{l \in L} l \rightarrow ES_{\Pi}(L)$*.* ⊠

2.1.7 ASP programs with variables

ASP tools [46, 73] admit the use of variables in the specifications. Typically a first order disjunctive logic program consists of a set of first-order rules of the form

$$\forall x_1, ..., x_s \, (\, a_1 \vee \cdots a_k \quad \leftarrow \quad b_1, \cdots, b_m, \, not \, b_{m+1}, \cdots, not \, b_n) \quad (2.22)$$

where every a_i and b_i are predicates whose variables belong to the list $x_1,\cdots,x_s$. We assume that the use of function symbols is not allowed. ASP tools use a different notation where universal quantifiers are omitted and the $\leftarrow$ symbol is replaced by the Prolog if, ‘‘`:-`’’. Following this notation, the rule[1]

$$a(x) \vee b(x) \leftarrow p(x)$$

is equivalent to the ASP rule

```
a(X) v b(X) :- p(X).
```

As an example of ASP program, Figure 2.3 shows a first-order representation of the well-known cabbage, goat and wolf puzzle, which is presented next:

Example 2.2 (Cabbage, goat and wolf puzzle)**.** *Once upon a time a farmer went to market and purchased a wolf, a goat, and a cabbage. On his way home, the farmer came to the bank of a river and rented a boat. But in crossing the river by boat, the farmer could carry only himself and a single one of his purchases - the wolf, the goat, or the cabbage. If left together, the wolf would eat the goat, or the goat would eat the cabbage. The farmer's challenge was to carry himself and his purchases to the far bank of the river.* ⊠

The ASP representation in Figure 2.3 contains four different constants representing the wolf `w`, the goat `g` and the cabbage `c` as well as the boat `b`. Action `m(X,T)` means that we move some item `w,g,c`, at time `T`, from one bank to the other (represented by the constants `r` and `l` respectively). We assume that the

[1]The usual notation in ASP, like in Prolog, represents variables by capital letters.

boat is always switching between both banks of the river, so when no action is executed, this means we moved the boat without carrying anything. We use a unique predicate fluent `at(Y,B,T)` meaning that `Y` is at bank `B` at a time `T`, being `Y` an item or the boat `b`.

Every first-order ASP program has associated a Herbrand domain (or simply domain), concept that is defined below:

Definition 2.9 (Herbrand domain)**.** *Given a first-order logic program* Π*, the* Herbrand domain *of* Π *(denoted by* D*) is the set of all terms that can be formed from constants and function symbols in* Π*.* ⊠

Since we do not allow the use of function symbols, the Herbrand domain is finite and corresponds to the set of constants of the program. For example, the domain of the ASP program of Figure 2.3 is the set $\{0, \cdots, max, l, r, w, g, c, b\}$.

Definition 2.10 (Ground instance)**.** *A ground instance of a rule* r*, which is of the form* (2.22)*, is any rule* r' *obtained from* r *by applying a substitution*

$$\theta : var(r) \rightarrow D$$

to the variables in r*, denoted as* $var(r)$*. For any rule* r*, we denote by* $Gr_D(r)$ *the set of all possible ground instances of* r*, and for any program* Π *let*

$$Gr_D(\Pi) \stackrel{\text{def}}{=} \bigcup_{r \in \Pi} Gr_D(r)$$

the so called grounding *of program* Π ⊠

Intuitively, $Gr_D(r)$ allows for the materialisation of the universal quantification of variables appearing in r. Roughly speaking, r is a shortcut denoting a set of rules $Gr_D(r)$. The range of each variable occurring in r is given by the set of terms appearing in D. Since D is finite, $Gr_D(\Pi)$ consists of a finite set of rules.

ASP tools usually compute the stable models of a first-order program Π as the stable models of its corresponding $Gr_D(\Pi)$ but this process is sound only if Π is domain independent.

Definition 2.11 (Domain independence)**.** *Let* Π *be a first-order ASP program and* D *its Herbrand domain.* Π *is said to be* domain independent *if given* $D' = D \cup \{w\}$*, where* w *is a fresh constant, it holds that* $Gr_D(\Pi)$ *has the same stable models as* $Gr_{D'}(\Pi)$*.* ⊠

This property guarantees that the stable models of a program with variables Π can be computed as the stable models of $Gr_D(\Pi)$ regardless of any domain $D' \supseteq D$. As an example, let us consider the program Π

$$q(1). \tag{2.23}$$

$$p \leftarrow not\ q(X). \tag{2.24}$$

This program is not domain independent. If we fix $D = \{1\}$, $Gr_D(\Pi)$ has $\{q(1)\}$ as the unique stable model. However, with $D = \{1, 2\}$ the stable model

```
1. time(0..max). opp(l,r). opp(r,l).
2. item(w). item(g). item(c).
3. eats(w,g). eats(g,c). object(b).
4. object(Z) :- item(Z).

5. % Effect axiom for moving
6. at(X,A,T+1):- at(X,B,T), m(X,T), opp(A,B).

7. % The boat is always moving
8. at(b,A,T+1):- at(b,B,T), opp(A,B).

9. % Inertia
10. at(Y,A,T+1) :- at(Y,A,T), not at(Y,B,T+1),  opp(A,B).

11. % State constraint
12. :- eats(X,Y), at(X,A,T), at(Y,A,T), at(b,B,T), opp(A,B).

13. % Unique value constraint
14. :- at(Y,A,T), at(Y,B,T), opp(A,B).

15. % Choice rules for action execution
16. m(X,T) :- not a(X,T), time(T), item(X).
17. a(X,T) :- not m(X,T), time(T), item(X).

18. % Action executability
19. :- m(X,T), at(b,A,T), at(X,B,T), opp(A,B).

20. % Non-concurrent actions
21. :- m(X,T), m(Z,T), X != Z.

22. % Initial state: everything at
23. % left bank
24. at(Y,l,0):- object(Y).
```

Figure 2.3: ASP encoding of Example 2.2

becomes $\{q(1), p\}$. Usually ASP tools impose a syntactic restriction, called *safety condition*, on the rules which guarantees domain independence. The must common (and the weakest known) safety condition for disjunctive programs is defined below:

Definition 2.12 (Safety condition from [15])**.** *An ASP rule is* safe *iff every variable that occurs in a rule (either in its head or in its body) also occurs in its positive body. A logic program* Π *is safe if all its rules are safe.* ⊠

An ASP *Grounder* is a program that transforms a safe ASP program Π with variables into its corresponding $Gr_D(\Pi)$. Unfortunately, $Gr_D(\Pi)$ may contain an extremely large number of ground rules. For instance, let us consider the program of Figure 2.3. If we fix the value of `max` to 2, D would correspond to the set {0, 1, 2, l, r, c, g, w, b} and, therefore, a brute-force grounder would ground the rule of inertia

```
at(Y,A,T+1) :- at(Y,A,T), not at(Y,B,T+1),opp(A,B).      (2.25)
```

into 59049 ground rules[2]. Most of these ground rules, like

```
at(c,a,1) :- at(c,a,0), not at(c,a,1),opp(a,a).      (2.26)
```

can be omitted since their bodies are false in every stable model. Of course the smaller the ground program, the higher performance is obtained. Grounders [46, 73] implement sophisticated algorithms (see Algorithm 1) that keep track of a set of ground atoms, DE, whose elements can potentially be true in at least one stable model. This set is used to restrict all the possible mappings of variables to constants of D to those, θ, that satisfy $body^+(r^\theta) \subseteq DE$, that is when the positive body of the rule r^θ, which comes from applying the mapping θ applied to a first-order rule r, is a subset of DE. This operation is performed until no ground rules are obtained. As an example of this improvement, if we feed Algorithm 1 with the program of Figure 2.3, it would translate the rule (2.25) into the 9 ground instances shown in Figure 2.4. Note that rules

[2]If we suppose that T and $T+1$ are two different variables with no relation between them, we can compute 9^5 possible substitutions.

like (2.26) are note generated by Algorithm 1.

Algorithm 1: Naive grounding algorithm

Require: An input (safe) logic program Π with variables
Ensure: A propositional ASP program $Gr_D(\Pi)$
that preserves the stable models
$DE := \emptyset$ {DE is the set of derivable facts}
$Gr_D(\Pi) := \emptyset$
repeat
  $DE' := DE$
  for all $r \in \Pi$ **do**
    $B := body^+(r)$
    $M = compute\_mappings(B, DE)$ {M are all possible mappings from predicates of $body^+(r)$ into derivable facts of DE}
    for all $\theta \in M$ **do**
      $DE := DE \cup head(r)$
      $Gr_D(\Pi) := Gr_D(\Pi) \cup \{r^\theta\}$
    end for
  end for
until $DE = DE'$
return $Gr_D(\Pi)$

```
at(c,l,1):-not at(c,r,1).
at(g,l,1):-not at(g,r,1).
at(w,l,1):-not at(w,r,1).
at(w,r,2):-at(w,r,1),not at(w,l,2).
at(g,r,2):-at(g,r,1),not at(g,l,2).
at(c,r,2):-at(c,r,1),not at(c,l,2).
at(w,l,2):-at(w,l,1),not at(w,r,2).
at(g,l,2):-at(g,l,1),not at(g,r,2).
at(c,l,2):-at(c,l,1),not at(c,r,2).
```

Figure 2.4: Ground instances of rule (2.25) generated by Algorithm 1.

Algorithm 1 can even be improved by using information from the dependency graph of the input program (see Section 2.1.1) to rearrange the order of evaluation of the rules in order to do less iterations.

2.1.8 General stable models

Apart from the classical definition of stable models, many other definitions have been proposed (thirteen different definitions of stable models are mentioned, for instance, in [77]). Any of them provides us with a different view of logic programs and such diversity allows us to choose among different alternatives depending on our convenience. One definition that we will conveniently use in this thesis is the so called *General Stable Models* [38], abbreviated as SM operator. It is a modification of the *Circumscription* ($CIRC$) operator defined

by John McCarthy [90], which is one of the most popular non-monotonic techniques that are present in the literature, deeply studied, extended and applied as well. The idea behind $CIRC$ consists in, given some classical theory Γ, *circumscribing* the extension of some predicate p exclusively to those facts explicitly asserted in Γ.

Given two predicates p and p', we define the following relations between them.

$$\begin{aligned} p' \leq p &\stackrel{\text{def}}{=} \forall X.\,(p'(X) \rightarrow p(X)) \\ p' = p &\stackrel{\text{def}}{=} \forall X.\,(p'(X) \leftrightarrow p(X)) \\ p' < p &\stackrel{\text{def}}{=} p' \leq p \wedge p' \neq p \end{aligned}$$

where X is a vector of variables. Those relations are used in the following definition of the operator $CIRC$.

Definition 2.13 (circumscription)**.** *Let Γ be a first-order theory and $\boldsymbol{p}$ a tuple of predicates. $CIRC\,[\Gamma;\boldsymbol{p}]$ is defined as the following second order formula:*

$$CIRC\,[\Gamma;\boldsymbol{p}] \stackrel{\text{def}}{=} \Gamma(\boldsymbol{p}) \wedge \neg\exists \boldsymbol{p'}.\,(\boldsymbol{p'} < \boldsymbol{p} \wedge \Gamma(\boldsymbol{p'}))\,.$$

⊠

Note that circumscription is a method to obtain non-monotonic consequences from a classical theory but it is not a non-monotonic logic by itself. By defining $CIRC$ as a second order formula, the idea behind becomes clear: fixing the minimal extent for $\boldsymbol{p}$ among all the possible predicates $\boldsymbol{p'}$ that satisfy Γ.

An alternative semantics was defined in [75]. Given two interpretations for a theory, M_1 and M_2, by $M_1 \leq^p M_2$ we mean that both M_1 and M_2 coincide in their universes and valuations for all constants and predicates with the only exception of p which satisfies $M_1[p] \subseteq M_2[p]$. Broadly speaking, circumscribing p in Γ would mean selecting the models with the minimal extension of p while leaving the rest fixed.

In a similar way to this definition of Circumscription, Ferraris, Lee and Lifschitz defined [38] the SM operator which captures exactly the stable models of an arbitrary first-order theory. This definition uses the translation from here and there to propositional logic defined in [100] and shown in 2.2.3 and extended to obtain a second order formula which captures the stable models of a theory Γ.

Definition 2.14 (SM operator from [38])**.** *Let Γ be a first-order theory. $SM\,[\Gamma]$ is defined as the following second-order formula:*

$$SM\,[\Gamma] \stackrel{\text{def}}{=} \Gamma \wedge \neg\exists \boldsymbol{p'} \text{ s.t. } (\boldsymbol{p'} < \boldsymbol{p} \wedge \Gamma^*(\boldsymbol{p'}))$$

where $\boldsymbol{p}$ stands for the list of all predicates occurring in Γ, $\boldsymbol{p'}$ is a list of fresh predicate variables (with the same length as $\boldsymbol{p}$) which do not occur in Γ and Γ^* is the next recursive translation[3]*:*

[3] In fact, F^* is an extension of the star-transformation defined in [100].

- $p_i(\boldsymbol{t}) = p_i'(\boldsymbol{t})$, *for any tuple* $\boldsymbol{t}$ *of terms with the same arity as* p_i*;*
- $F^* = F$*, for any atomic formula* F *that does not contain members*[4] *of* $\boldsymbol{p}$*;*
- $(F \wedge G)^* = F^* \wedge G^*$*;*
- $(F \vee G)^* = F^* \vee G^*$*;*
- $(F \rightarrow G)^* = (F \rightarrow G) \wedge (F^* \rightarrow G^*)$*;*
- $(\forall x F)^* = \forall x F^*$*;*
- $(\exists x F)^* = \exists x F^*$.

⊠

As stated below, when the syntax is restricted to the case of normal logic programs, this operator becomes a generalisation of the traditional definition of stable models [48] to non-Herbrand models. It is shown in the following proposition.

Proposition 2.2 (from [38])**.** *Let* At *be a propositional signature containing at least one object constant, and* Π *a finite set of rules of the form*

$$A_0 \leftarrow A_1, \cdots, A_m, not\ A_{m+1}, \ldots, A_n, \tag{2.27}$$

where $A_1, \cdots A_n$ *are atomic formulas of* At *not containing equality. For any set* X *of ground terms of* At*, the following conditions are equivalent.*

1. X *is a stable model of* Π *in the sense of [48];*
2. *the Herbrand interpretation of* At *that makes the elements of* X *true and all other ground atoms false is a stable model of the formula corresponding to* Π*.*

⊠

Moreover, a more general result has been proven:

Theorem 2.3 (from [38])**.** *A model of a first order theory* F *is stable if it satisfies* $SM\,[F]$. ⊠

SM operator has been studied in detail proving several useful properties like the one we present next, which will be used later.

Proposition 2.3 (from [38])**.** *If a formula* γ *starts with* $\neg$ *then the formula*

$$\overline{p'} \leq \overline{p} \rightarrow (\gamma^*(p') \leftrightarrow \gamma)$$

is logically valid. ⊠

In order to show how the SM operator can be applied, let us consider the following logic program Π from [38]:

[4] It includes the case when $F = \bot$.

$$
\begin{array}{lcl}
p(a). & & \\
q(b). & & \\
r(x) & \leftarrow & p(x) \wedge not\ q(x).
\end{array}
$$

whose unique stable model is $\{p(a), q(b)\}$. Before applying the SM operator we must rewrite Π as a theory Γ in logical notation as follows:

$$\Gamma = p(a) \wedge q(b) \wedge \forall x\ \ (p(x) \wedge \neg q(x) \rightarrow r(x))$$

Now, if we use the SM operator we obtain the following second order formula

$$
\begin{array}{rcl}
SM\,[\Gamma] & \equiv & p(a) \wedge q(b) \wedge \forall x\,(p(x), \neg q(x) \rightarrow r(x)) \\
& \wedge & \neg\exists\,(p', q,' r')\ \Big((p', q', r') < (p, q, r) \\
& \wedge & p'(a) \wedge p'(b) \wedge \underbrace{\forall x\,(p(x) \wedge \neg q(x) \rightarrow r(x))}_{\text{(A)}} \\
& \wedge & \forall x(p'(x) \wedge \underbrace{\neg q(x) \wedge \neg q'(x)}_{\text{(B)}} \rightarrow r'(x))\Big)
\end{array}
$$

Note that the subformula (A) can be taken out of the scope of the existential quantifier and then simplified by means of several laws of classical logic. On the other hand, because of Proposition 2.3, the subformula (B) is equivalent to $\neg q(x)$. The simplified formula would be:

$$
\begin{array}{rcl}
SM\,[\Gamma] & \equiv & p(a) \wedge q(b) \wedge \forall x\,(p(x), \neg q(x) \rightarrow r(x)) \\
& \wedge & \neg\exists\,(p', q,' r')\ \Big((p', q', r') < (p, q, r) \\
& \wedge & p'(a) \wedge p'(b) \wedge \forall x\ \ (p'(x) \wedge \neg q(x) \rightarrow r'(x))\Big)
\end{array}
$$

Although this example of second order formula can be reduced to a first-order sentence removing the second order quantifiers, in the general case, however, this is not always possible. For instance, it is well-known that the problem of graph reachability can be expressed in terms of a logic program with variables (and so, by using the SM operator) but not with a first-order theory (the proof can be found in [74]).

2.1.9 Computational complexity

In this section, we briefly summarise complexity results for the basic decision problems in ASP. In what follows, let a be an atom and X be a set of atoms.

- For a positive normal logic program Π

  - Deciding whether X is the stable model of Π is P-COMPLETE.
  - Deciding whether a belongs to a stable model of Π is P-COMPLETE.
- For a normal logic program Π
  - Deciding whether X is a stable model of Π is P-COMPLETE.
  - Deciding whether a belongs to a stable model of Π is NP-COMPLETE.
- For a positive disjunctive logic program Π
  - Deciding whether X is an stable model of Π is CONP-COMPLETE.
  - Deciding whether a belongs to a stable model of Π is Σ_2^p-COMPLETE.
- For a propositional theory Γ
  - Deciding whether X is an stable model of Γ is CONP-COMPLETE.
  - Deciding whether a belongs to a stable model of Γ is Σ_2^p-COMPLETE.

The above complexity results apply to propositional programs only. For capturing the complexity of the first-order case, we note that $Gr_D(\Pi)$ of a first order program Π is exponential in the size of Π. Hence, roughly speaking, we obtain the analogous results by replacing the base NP by NEXPTIME for capturing programs with variables.

2.2 Equilibrium logic

The original definition of stable models was restricted to the logic programming syntax and could not be applied to arbitrary logical theories. Furthermore, as we have seen, semantics depended on a syntactic transformation (the program reduct).

Based on a characterisation of the monotonic logic of HT [57] and a model selection criterion, *Equilibrium Logic* [98, 99] has become the most successful logical characterisation of stable models and ASP. We recall now its basic definitions and results.

2.2.1 Syntax and semantics

Given a set of atoms At, an HT *interpretation* is defined as the structure $\langle H, T\rangle$, composed by two sets of atoms, H and T, respectively standing for two worlds "here" and "there" such that $H \subseteq T \subseteq At$ (interpertations in which $H = T$ are called *total interpretations*).

Definition 2.15 (HT satisfaction from [99])**.** *Given an HT interpretation $\langle H, T\rangle$ and a formula φ with signature At, the satisfaction rules are defined in a recursive way as follows:*

- $\langle H, T\rangle \not\models \bot$
- $\langle H, T\rangle \models p$ *iff* $p \in H$*, for any atom* $p \in At$

- $\langle H,T\rangle \models \varphi \vee \psi$ *iff* $\langle H,T\rangle \models \varphi$ *or* $\langle H,T\rangle \models \psi$
- $\langle H,T\rangle \models \varphi \wedge \psi$ *iff* $\langle H,T\rangle \models \varphi$ *and* $\langle H,T\rangle \models \psi$
- $\langle H,T\rangle \models \varphi \rightarrow \psi$ *iff* $(\langle H,T\rangle \not\models \varphi$ *or* $\langle H,T\rangle \models \psi)$ *and* $(\langle T,T\rangle \models \varphi \rightarrow \psi)$.

⊠

Other usual connectives can be defined in terms of the previous ones. For instance, default negation can be regarded as a shorthand of implication, $\neg\varphi \stackrel{\text{def}}{=} \varphi \rightarrow \bot$.

The logic of HT is actually equivalent to the so-called Gödels logic G_3 [52]. G_3 is defined by a multivalued function M in terms of the values {0,1,2}. The intuition behind this correspondence is that, given an HT interpretation $\langle H,T\rangle$ with $H \subseteq T$, we can have three possible situations for any atom p: $p \in H$ (the atom is true or 2), $p \notin T$ (the atom is false or 0) or $p \in T \setminus H$ (the atom is undefined or 1).

Definition 2.16 (valuation of formulas from [99])**.** *Given an HT formula φ and an HT interpretation $\langle H,T\rangle$, we define the corresponding* valuation M *as a function*

$$\mathbf{M} : \mathcal{L} \rightarrow \{0,1,2\}$$

from the set of formulas $\mathcal{L}$ to $\{0,1,2\}$ and specified as follows:

1. $\mathbf{M}(p) \stackrel{\text{def}}{=} \begin{cases} 2 & \text{if } p \in H \\ 0 & \text{if } p \notin T \\ 1 & \text{if } p \in T \setminus H \end{cases}$
2. $\mathbf{M}(\varphi \wedge \psi) \stackrel{\text{def}}{=} min(\mathbf{M}(\varphi), \mathbf{M}(\psi))$;
3. $\mathbf{M}(\varphi \vee \psi) \stackrel{\text{def}}{=} max(\mathbf{M}(\varphi), \mathbf{M}(\psi))$
4. $\mathbf{M}(\varphi \rightarrow \psi) \stackrel{\text{def}}{=} \begin{cases} 2 & \text{if } \mathbf{M}(\varphi) \leq \mathbf{M}(\psi) \\ \mathbf{M}(\psi) & \text{otherwise} \end{cases}$

⊠

Several useful properties of HT are described next:

Proposition 2.4 (properties of HT)**.** *Given an HT formula φ with signature At and an HT interpretation $\langle H,T\rangle$, then the following properties hold:*

i) *If $\langle H,T\rangle \models \varphi$ then $\langle T,T\rangle \models \varphi$;*

ii) *$\langle T,T\rangle \models \varphi$ iff $T \models \varphi$ in classical logic;*

iii) *$\langle H,T\rangle \models \neg\varphi$ iff $T \not\models \varphi$.*

⊠

Property i) is known as *persistence* in intuitionistic terminology and allows showing that every formula valid in HT is also valid in classical logic. Properties ii) and iii) respectively state that every total model is also a classical model and that default negation can be evaluated in classical logic using the T-component.

The model selection criterion that induces the nonmonotonic behaviour is defined as follows:

Definition 2.17 (equilibrium model from [99])**.** *Every HT total interpretation $M = \langle T, T\rangle$ is an equilibrium model of a formula φ iff $M \models \varphi$ and there is no interpretation $\langle H, T\rangle$, with $H \subset T$ such that $\langle H, T\rangle \models \varphi$.* ⊠

The definition of equilibrium model generalises the concept of stable model.

Theorem 2.4 (from [99])**.** *A total HT interpretation $\langle T, T\rangle$ is an equilibrium model of a theory Γ iff T is a stable model of Γ.* ⊠

2.2.2 Normal forms for Here and There

Cabalar and Ferraris [22] proposed a *conjunctive normal form* (CNF) and a *disjunctive normal form* (DNF) for here and there theories. Similarly to the CNF for classical logic, a here and there theory Γ can be translated into CNF, which consists of a set of implications that are defined next, starting from its set of countermodels.

Definition 2.18 (from [22])**.** *Given an HT interpretation $\langle H, T\rangle$ under some propositional signature At, we define $\psi_{\langle H,T\rangle}$ as the (nonnested) rule:*

$$\psi_{\langle H,T\rangle} \stackrel{\text{def}}{=} \left(\bigwedge_{a\in H} a\right) \wedge \left(\bigwedge_{b\in At\setminus T} \neg b\right) \rightarrow \left(\bigvee_{c\in T\setminus H} (c \vee \neg c)\right).$$

⊠

For instance, if $At = \{p, q, r\}$, $H = \{q\}$ and $T = \{p, q\}$, $\psi_{\langle H,T\rangle}$ corresponds to the rule

$$q \wedge \neg r \rightarrow p \vee \neg p.$$

Note that, in case we deal with total interpretations, the consequent of the implication would be equal to $\bot$. The following result shows that an arbitrary HT theory Γ can be translated into a conjunction of implications, one for each countermodel of Γ:

Theorem 2.5 (Conjunctive normal form from [22])**.** *Let Γ be an HT theory over a finite signature At and let $F(\Gamma)$ denote a formula constructed from the countermodels of Γ:*

$$F(\Gamma) \stackrel{\text{def}}{=} \bigwedge_{\langle H,T\rangle : \langle H,T\rangle \not\models \Gamma} \psi_{\langle H,T\rangle}.$$

Then Γ is equivalent to $F(\Gamma)$ under the logic of here and there. ⊠

Analogously to the CNF, the DNF of a theory Γ comes from its set of models. Every disjunct has the following form:

Definition 2.19 (from [22]). *Given an HT interpretation $\langle H, T\rangle$, for a finite signature At, let $\varphi_{\langle H,T\rangle}$ be the formula*

$$\varphi_{\langle H,T\rangle} \stackrel{\text{def}}{=} \left(\bigwedge_{a\in H} a\right) \wedge \left(\bigwedge_{b\in At\setminus T} \neg b\right) \wedge \left(\bigwedge_{c\in T\setminus H} \neg\neg c\right) \wedge \left(\bigwedge_{d,e\in T\setminus H} (d \to e)\right).$$

⊠

For instance, if $At = \{p, q, r\}$, $H = \{q\}$ and $T = \{p, q\}$, then $\varphi_{\langle H,T\rangle}$ corresponds to

$$q \wedge \neg r \wedge \neg\neg p \wedge (p \to p)\,.$$

The interesting properties of $\varphi_{\langle H,T\rangle}$ are stated in the next proposition and theorem:

Proposition 2.5 (from [22]). *The only interpretations that satisfy $\varphi_{\langle H,T\rangle}$ are $\langle H, T\rangle$ and also $\langle T, T\rangle$.* ⊠

Theorem 2.6 (Disjunctive normal form from [22]). *Let Γ be an HT theory over a finite signature At and let $F(\Gamma)$ denote a formula constructed from the models of Γ:*

$$F(\Gamma) \stackrel{\text{def}}{=} \bigvee_{\langle H,T\rangle:\langle H,T\rangle\models\Gamma} \varphi_{\langle H,T\rangle}.$$

Then Γ is equivalent to $F(\Gamma)$ under the logic of here and there. ⊠

2.2.3 Translating equilibrium logic into propositional logic

Every HT formula φ built over a signature At can be translated into classical propositional logic by introducing a new atom p' for each original atom $p \in At$. Every original atom p represents its truth value in the T world whereas the new introduced atom p' will play the role of p at H. Furthermore, the condition $H \subseteq T$ is captured by the inclusion of the following expression:

$$\bigwedge_{p\in At} p' \to p.$$

The translation into propositional logic is defined as follows:

Definition 2.20 (from [100]). *Given an HT formula φ, the propositional formula $\varphi*$ is defined as follows:*

- $(\bot)^* \stackrel{\text{def}}{=} \bot$
- $(p)^* \stackrel{\text{def}}{=} p'$

- $(\varphi \wedge \psi)^* \stackrel{\text{def}}{=} (\varphi)^* \wedge (\psi)^*$
- $(\varphi \vee \psi)^* \stackrel{\text{def}}{=} (\varphi)^* \vee (\psi)^*$
- $(\varphi \rightarrow \psi)^* \stackrel{\text{def}}{=} (\varphi \rightarrow \psi) \wedge (\varphi^* \rightarrow \psi^*)$

⊠

2.2.4 Strong equivalence

In NMR, the regular equivalence, understood as a mere coincidence of selected models, is too weak to consider that one theory Γ_1 can be safely replaced by a second one Γ_2 since the addition of a context Γ may make them behave in a different way due to non-monotonicity. Therefore, we need to consider a stronger notion of equivalence, which is called *strong equivalence* and is formally defined next:

Definition 2.21 (Strong Equivalence from [78])**.** *We say that two theories Γ_1 and Γ_2 are* strongly equivalent *when, for any arbitrary theory Γ, both $\Gamma_1 \cup \Gamma$ and $\Gamma_2 \cup \Gamma$ have the same selected models (in this case, stable models).* ⊠

Later, in [78], authors proved that checking equivalence in the logic of Here-and-There is a necessary and sufficient condition for strong equivalence in Equilibrium Logic, that is:

Lemma 2.2 (strong squivalence from [78])**.** *Two theories Γ_1 and Γ_2 are strongly equivalent iff $\Gamma_1 \equiv_{HT} \Gamma_2$.* ⊠

The notion of strong equivalence has been deeply studied in the propositional case (in fact, some strong equivalence checkers are, for instance, [116] and [26] respectively based on HT-tableaux and the translation into SAT proposed in [83]) and then this result for propositional HT was further extended to arbitrary first-order theories in [79]. It must be noticed that one direction of this result, the sufficient condition, is actually trivial. As HT is monotonic, $\Gamma_1 \equiv_{HT} \Gamma_2$ implies $\Gamma_1 \cup \Gamma \equiv_{HT} \Gamma_2 \cup \Gamma$ and so, their selected models will also coincide. The real significant result is the opposite direction, namely, that HT-equivalence is also a necessary condition for strong equivalence, as it shows that HT is strong enough as a monotonic basis for Equilibrium Logic.

2.2.5 Quantified equilibrium logic

Having been Equilibrium Logic conceived as a propositional formalism, it has been extended to first order theories [101], becoming an important framework to prove properties of logic programs with variables [79].

The new formalism, called *Quantified Equilibrium Logic* (QEL), allows ASP first-order programs to be simplified and, sometimes, partially simplified before grounding. Moreover, its monotic basis, *Quantified Here and There* (QHT), can be used to check the property of strong equivalence, as happens with HT in the propositional case.

The definition of QHT is based on a first order language denoted by $\mathcal{L} = \langle C, F, P \rangle$, where C, F and P are three disjoint sets that represent constants,

functions and predicates respectively. Given a domain D the sets $At_D(C, P)$ and $At_D(C, F)$ are defined as follows:

- $At_D(C, P)$ stands for all atomic instances that can be formed from $\langle C \cup D, F, P\rangle$.
- $T_D(C, F)$ all ground terms that can be obtained from $\langle C \cup D, F, P\rangle$.

A QHT interpretation is a tuple $\mathcal{M} = \langle (D_h, D_t, \sigma), I_h, I_t\rangle$ such that

- D_h is the domain for here and D_t the *expanded* domain for there. It holds that $D_h \subseteq D_t$.
- $\sigma : T_{D_t}(C, F) \to D_t$ is a mapping from ground terms into elements of the expanded domain satisfying that $\sigma(d) = d$ if $d \in D_t$ and $\sigma(\tau) \in D_h$ if $\tau \in D_h(C, F)$,
- $I_h \subseteq At_{D_h}(\emptyset, \emptyset, P)$, $I_h \subseteq At_{D_t}(\emptyset, \emptyset, P)$ are two sets of ground atoms such that $I_h \subseteq I_t$

Along this section, we deal with *static* domains and *decidable equality*, which means that $D_h = D_t$ and that equality predicate is interpreted with the same condition in both worlds. Therefore our QHT structures are of the form $\langle (D_t, \sigma), I_h, I_t\rangle$ which, intuitively, means that $\mathcal{M}$ can be seen as a first order interpretation with two components: here and there in the sense of Kripke semantics for Intuitionistic Logic [30], where worlds are ordered by $h \leq t$.

Definition 2.22 (QHT semantics from [101])**.** *The satisfaction relation for a QHT interpretation $\mathcal{M} = \langle (D, \sigma), I_h, I_t\rangle$ is defined as follows:*

- $\mathcal{M} \models \top, \quad \mathcal{M} \not\models \bot$
- $\mathcal{M} \models p(\tau_1, \cdots, \tau_n)$ *iff* $p(\sigma(\tau_1), \cdots, \sigma(\tau_n)) \in I_h$
- $\mathcal{M} \models \tau = \tau'$ *iff* $\sigma(\tau) = \sigma(\tau')$.
- $\mathcal{M} \models \varphi \wedge \psi$ *iff* $\mathcal{M} \models \varphi$ *and* $\mathcal{M} \models \psi$
- $\mathcal{M} \models \varphi \vee \psi$ *iff* $\mathcal{M} \models \varphi$ *or* $\mathcal{M} \models \psi$
- $\mathcal{M} \models \varphi \to \psi$ *iff* $\mathcal{M} \not\models \varphi$ *or* $\mathcal{M} \models \psi$*, and* $\langle (D, \sigma), I_t, I_t\rangle \models \varphi \to \psi$
- $\mathcal{M} \models \forall\, x, \varphi(x)$ *iff* $\mathcal{M} \models \varphi(d)$*, for all* $d \in D$
- $\mathcal{M} \models \exists\, x, \varphi(x)$ *iff* $\mathcal{M} \models \varphi(d)$*, for some* $d \in D$

⊠

Given two QHT interpretations, $\mathcal{M} = \langle (D, \sigma), I_h, I_t\rangle$ and $\mathcal{M}' = \langle (D', \sigma'), I'_h, I'_t\rangle$, we say that $\mathcal{M} \leq \mathcal{M}'$ iff $D = D'$, $\sigma = \sigma'$, $T = T'$ and $H \subseteq H'$. If, additionally, $H \subset H'$ we say that the relation is strict (denoted by $\mathcal{M} < \mathcal{M}'$).

Definition 2.23 (quantified equilibrium model from [101])**.** *Let φ be a QHT formula. A QHT total interpretation $\mathcal{M}$ is a first-order equilibrium model of φ if $\mathcal{M} \models \varphi$ and there is no model $\mathcal{M}' < \mathcal{M}$ of φ.* ⊠

Several interesting properties arise from the study of Quantified Equilibrium Logic. For instance, QHT is a suitable monotonic basis for checking the property of strong equivalence presented, for the propositional case, in Section 2.2.4. If we consider first-order theories we get the following result:

Theorem 2.7 (first-order strong equivalence (from [79]))**.** *A set Γ of first-order sentences is strongly equivalent to a set Δ if both theories are equivalent in SQHT$^=$.* ⊠

Another interesting property, proved by Ferraris, Lee and Lifschitz in [38], is the relation between Quantified Equilibrium Logic and the General Theory of Stable Models (see Section 2.1.8):

Theorem 2.8 (from [38])**.** *Given a first-order formula φ and a tuple of predicates $\boldsymbol{p}$, any interpretation $\mathcal{M}$ is a model of SM$[\varphi;\boldsymbol{p}]$ iff $\mathcal{M}$ is a stable model (in the sense of Quantified Equilibrium Logic) of φ.* ⊠

2.2.6 Infinitary equilibrium logic

The use of Infinitary Formulas [109, 64] has been firstly adapted to Logic Programming by Mirosław Truszczyński [115]. This formalism is able to deal with infinite domains, like the case of temporal programs, using only three different connectives. Those connectives are $\{\}^\wedge$, which stands for an infinite conjunction, $\{\}^\vee$, which is a shorthand of an infinite disjunction and, finally, $\rightarrow$ which is equivalent to the implication used in logic programming.

Definition 2.24 (syntax from [115])**.** *Let At be a* propositional signature *(we assume the existence of a constant $\bot$, different for all symbols in At that plays the role of* falsity*). We define the sets $\mathcal{F}_0, \mathcal{F}_1, \cdots$ by induction as follows:*

1. $\mathcal{F}_0^{At} = At \cup \{\bot\}$
2. *$\mathcal{F}_{i+1}^{At}$ is obtained from $\mathcal{F}_i^{At}$ by adding expressions $\mathcal{H}^\wedge$ and $\mathcal{H}^\vee$ for all subsets $\mathcal{H}$ of $\mathcal{F}_i^{At}$, and expressions $F \rightarrow G$ for all $F, G \in \mathcal{F}_i^{At}$.*

The elements of $\bigcup_{i=0}^{\infty} \mathcal{F}_i^{At}$ are called (infinitary) formulas over At. ⊠

The rest of the connectives can be easily defined in terms of these ones. For instance, $G \wedge F \equiv \{F, G\}^\wedge$, $G \vee F \equiv \{F, G\}^\vee$, $\neg F \equiv F \rightarrow \bot$, $G \leftrightarrow F \equiv (F \rightarrow G) \wedge (G \rightarrow F)$ and $\top \equiv \neg\bot$, as happens in Here and There. Although the stable model semantics for infinitary formulas was formerly defined in terms of infinitary reducts, in [55] a new semantics, based on Here and There, was defined.

Definition 2.25 (from [55])**.** *We define an HT-interpretation on a (possibly infinite) signature At as a pair of sets of atoms, which we denote by $M = \langle H, T\rangle$, such that $H \subseteq T \subseteq At$. As happens in the finitary case, when $H = T$ we say that M is total. The satisfaction of formulas is defined as follows:*

- *$M \models p$ if $p \in H$, with $p \in At$.*
- *$M \models \mathcal{H}^\wedge$ if for every $F \in \mathcal{H}$, $M \models F$.*

- $M \models \mathcal{H}^\vee$ *if there exists* $F \in \mathcal{H}$ *such that* $M \models F$.
- $M \models F \to G$ *if either* $M \not\models F$ *or* $M \models G$ *and* $\langle T, T\rangle \models F \to G$.

⊠

The minimization criterion for selecting the equilibrium models is the same as in the propositional case. Formally, the definition of infinitary equilibrium model is given next.

Definition 2.26 (from [55])**.** *A Here and There interpretation* $\langle H, T\rangle$ *is an equilibrium model of an infinitary formula* $\mathcal{F}$ *if* $H = T$ *and there is no* $H' \subset T$ *such that* $\langle H', T\rangle \models \mathcal{F}$ *.* ⊠

Several equivalences in Here and There can also be extended to the infinitary case. For instance, we prove next two equivalences that will be used in Section 7.1.

Proposition 2.6. *The following equivalences are valid in Infinitary Here and There*

$$\mathcal{H}^\vee \to \beta \quad \Leftrightarrow \quad \{\alpha \to \beta \mid \alpha \in \mathcal{H}\}^\wedge \tag{2.28}$$
$$\alpha \to \mathcal{H}^\wedge \quad \Leftrightarrow \quad \{\alpha \to \beta \mid \beta \in \mathcal{H}\}^\wedge \tag{2.29}$$

⊠

Proof.

$$\begin{aligned}
\langle H, T\rangle \models \mathcal{H}^\vee \to \beta \quad &\Leftrightarrow \quad \begin{array}{l}\langle H, T\rangle \not\models \mathcal{H}^\vee \text{ or } \langle H, T\rangle \models \beta \\ \text{and} \\ \langle T, T\rangle \not\models \mathcal{H}^\vee \text{ or } \langle T, T\rangle \models \beta\end{array} && \text{(Def. 2.25)} \\
&\Leftrightarrow \quad \begin{array}{l}\forall \alpha \in \mathcal{H},\ \langle H, T\rangle \not\models \alpha \text{ or } \langle H, T\rangle \models \beta \\ \text{and} \\ \forall \alpha \in \mathcal{H},\ \langle T, T\rangle \not\models \alpha \text{ or } \langle T, T\rangle \models \beta\end{array} && \text{(Def. 2.25)} \\
&\Leftrightarrow \quad \begin{array}{l}\forall \alpha \in \mathcal{H},\ \langle H, T\rangle \models \alpha \to \beta \\ \text{and} \\ \forall \alpha \in \mathcal{H},\ \langle T, T\rangle \models \alpha \to \beta\end{array} \\
&\Leftrightarrow \quad \langle H, T\rangle \models \{\alpha \to \beta \mid \alpha \in \mathcal{H}\}^\wedge && \text{(Def. 2.25)}
\end{aligned}$$

$$\begin{aligned}
\langle H, T\rangle \models \alpha \to \mathcal{H}^\wedge \quad &\Leftrightarrow \quad \begin{array}{l}\langle H, T\rangle \not\models \alpha \text{ or } \langle H, T\rangle \models \mathcal{H}^\wedge \\ \text{and} \\ \langle T, T\rangle \not\models \alpha \text{ or } \langle T, T\rangle \models \mathcal{H}^\wedge\end{array} && \text{(Def. 2.25)} \\
&\Leftrightarrow \quad \begin{array}{l}\langle H, T\rangle \not\models \alpha \text{ or } \forall \beta \in \mathcal{H},\ \langle H, T\rangle \models \beta \\ \text{and} \\ \langle T, T\rangle \not\models \alpha \text{ or } \forall \beta \in \mathcal{H},\ \langle T, T\rangle \models \beta\end{array} && \text{(Def. 2.25)}
\end{aligned}$$

$$
\begin{array}{ll}
 & \forall \beta \in \mathcal{H},\ \langle H,T\rangle \not\models \alpha \text{ or } \langle H,T\rangle \models \beta \\
\Leftrightarrow & \text{and} \\
 & \forall \beta \in \mathcal{H},\ \langle T,T\rangle \not\models \alpha \text{ or } \langle T,T\rangle \models \beta \\
 & \\
 & \forall \beta \in \mathcal{H},\ \langle H,T\rangle \models \alpha \rightarrow \beta \\
\Leftrightarrow & \text{and} \\
 & \forall \beta \in \mathcal{H},\ \langle T,T\rangle \models \alpha \rightarrow \beta \\
 & \\
\Leftrightarrow & \langle H,T\rangle \models \{\alpha \rightarrow \beta \mid \beta \in \mathcal{H}\}^{\wedge} \qquad \text{(Def. 2.25)}
\end{array}
$$

⊠

2.3 Linear temporal logic

Linear Temporal Logic is a kind of *Temporal Logic* whose interpretations are limited to transitions which are discrete, reflexive, transitive, linear and total. For defining the LTL syntax, we start from a finite set of atoms At called the *propositional signature*. A (temporal) *formula* φ is defined by the grammar:

$$\varphi ::= \bot \mid p \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \bigcirc\varphi_1 \mid \Box\varphi_1 \mid \Diamond\varphi_1 \mid \varphi_1 \mathcal{U} \varphi_2 \mid \varphi_1 \mathcal{R} \varphi_2 \mid (\varphi_1)$$

where φ_1 and φ_2 are temporal formulas in their turn and p is an atom. Operator $\bigcirc$ is understood as "at the next moment", $\Box$ is read "forever" and $\Diamond$ stands for "eventually" or "at some future point. Operators $\mathcal{U}$ and $\mathcal{R}$, called "until" and "release", must be read as "φ_1 is true until φ_2 becomes true" and "φ_2 is true while its truth value is not released by φ_1. Semantically, temporal formulas are defined in terms of Kripke structures [68].

Definition 2.27 (semantics)**.** *Let At be a set of propositional variables. An LTL model is a Kripke structure of the form*

$$\langle W, R, V\rangle$$

where

- *W is a set of Kripke points, which intuitively corresponds to our set of* moments *in time (our accessible worlds or states),*
- *R is a reflexive, transitive, and linear accessibility relation, and*
- *$V : W \rightarrow 2^{At}$ is a mapping between a state and the set of propositional atoms that are true in it.*

However, since LTL has a linear, discrete basis that is isomorphic to $\mathbb{N}$, this model structure is often simplified from the above to

$$\langle \mathbb{N}, V\rangle$$

where $V : \mathbb{N} \rightarrow 2^{At}$ maps each Natural Number (representing a moment in time) to the set of propositions that are true at that moment. Based on the latter, let $\mathbf{M} = \langle \mathbb{N}, V\rangle$ and k be an LTL model and an integer respectively. The satisfaction relation (denoted by $\models$) is recursively defined as follows:

1. $\mathbf{M}, k \models p$ *iff* $p \in V(k)$*, for* $p \in At$.
2. $\mathbf{M}, k \models \neg\varphi$ *iff* $\mathbf{M}, k \not\models \varphi$.
3. $\mathbf{M}, k \models \varphi \wedge \psi$ *iff* $\mathbf{M}, k \models \varphi$ *and* $\mathbf{M}, k \models \psi$.
4. $\mathbf{M}, k \models \varphi \vee \psi$ *iff* $\mathbf{M}, k \models \varphi$ *or* $\mathbf{M}, k \models \psi$.
5. $\mathbf{M}, k \models \bigcirc\varphi$ *iff* $\mathbf{M}, k+1 \models \varphi$.
6. $\mathbf{M}, k \models \varphi \,\mathcal{U}\, \psi$ *iff there is* $j \geq k$ *such that* $\mathbf{M}, j \models \psi$ *and for all* $i \in [k, j-1]$, $\mathbf{M}, i \models \varphi$.
7. *never* $\mathbf{M}, k \models \bot$.

⊠

The rest of derived operators can be defined in terms of $\mathcal{U}$, leading to the expressions $\varphi \,\mathcal{R}\, \psi = \neg(\neg\varphi \mathcal{U} \neg\psi)$, $\Diamond\varphi = \top \,\mathcal{U}\, \varphi$ and $\Box\varphi = \bot \,\mathcal{R}\, \varphi$. A formula φ is *LTL-valid* if $\mathbf{M}, 0 \models \varphi$ for any $\mathbf{M}$. An LTL interpretation $\mathbf{M}$ is a *model* of a theory Γ, written $\mathbf{M} \models \Gamma$, if $\mathbf{M}, 0 \models \varphi$, for all formula $\varphi \in \Gamma$.

Proposition 2.7. *The following* equivalences *are valid in LTL*

$$\bigcirc(\varphi \vee \psi) \;\Leftrightarrow\; \bigcirc\varphi \vee \bigcirc\psi \tag{2.30}$$
$$\bigcirc(\varphi \wedge \psi) \;\Leftrightarrow\; \bigcirc\varphi \wedge \bigcirc\psi \tag{2.31}$$
$$\bigcirc(\varphi \mathcal{U} \psi) \;\Leftrightarrow\; (\bigcirc\varphi)\,\mathcal{U}\,(\bigcirc\psi) \tag{2.32}$$
$$\Diamond(\varphi \vee \psi) \;\Leftrightarrow\; \Diamond\varphi \vee \Diamond\psi \tag{2.33}$$
$$\Box(\varphi \wedge \psi) \;\Leftrightarrow\; \Box\varphi \wedge \Box\psi \tag{2.34}$$
$$\rho\,\mathcal{U}\,(\varphi \vee \psi) \;\Leftrightarrow\; (\rho\,\mathcal{U}\varphi) \vee (\rho\,\mathcal{U}\psi) \tag{2.35}$$
$$(\varphi \wedge \psi)\,\mathcal{U}\rho \;\Leftrightarrow\; (\varphi\,\mathcal{U}\rho) \wedge (\psi\,\mathcal{U}\rho) \tag{2.36}$$
$$\neg\bigcirc\varphi \;\Leftrightarrow\; \bigcirc\neg\varphi \tag{2.37}$$
$$\neg\Box\varphi \;\Leftrightarrow\; \Diamond\neg\varphi \tag{2.38}$$
$$\neg\Diamond\varphi \;\Leftrightarrow\; \Box\neg\varphi \tag{2.39}$$
$$\Box\varphi \;\Leftrightarrow\; \varphi \wedge \bigcirc\Box\varphi \tag{2.40}$$
$$\Diamond\varphi \;\Leftrightarrow\; \varphi \vee \bigcirc\Diamond\varphi \tag{2.41}$$
$$\varphi\mathcal{U}\psi \;\Leftrightarrow\; \psi \vee (\varphi \wedge \bigcirc(\varphi\mathcal{U}\psi)) \tag{2.42}$$
$$\varphi\mathcal{R}\psi \;\Leftrightarrow\; \psi \wedge (\varphi \vee \bigcirc(\varphi\mathcal{R}\psi)) \tag{2.43}$$

⊠

Apart from the formal definition we have given in this section, there are several alternative formalisations providing different, and interesting, ways to view LTL. We will focus our attention on two well-known characterisations: ω-languages and first-order logic.

2.3.1 LTL as a fragment of first-order logic

We now provide an alternative view of the semantics for LTL but, this time in first-order logic. Primarily this is achieved by representing temporal propositions as classical predicates parametrized by the moment in time being

considered [63]. The resulting first-order fragment is called *Monadic First-order logic of linear order* (MFO($<$)), which is characterised by the following conditions:

- Only one free variable is considered in any formula or subformula.
- There exists a predefined linear relation, denoted by the binary predicate $\leq$, among all first-order variables.
- The rest of predicates are monadic.

Next definition shows how an LTL formula can be translated into first-order logic

Definition 2.28. *Let φ be a temporal formula for signature At. We define the translation $[\varphi]_t$ for some time point $t \in \mathbb{N}$ as follows:*

$$\begin{aligned}
[\bot]_t &\stackrel{\text{def}}{=} \bot \\
[p]_t &\stackrel{\text{def}}{=} p(t), \text{ with } p \in At. \\
[\neg\alpha]_t &\stackrel{\text{def}}{=} \neg[\alpha]_t \\
[\alpha \wedge \beta]_t &\stackrel{\text{def}}{=} [\alpha]_t \wedge [\beta]_t \\
[\alpha \vee \beta]_t &\stackrel{\text{def}}{=} [\alpha]_t \vee [\beta]_t \\
[\alpha \rightarrow \beta]_t &\stackrel{\text{def}}{=} [\alpha]_t \rightarrow [\beta]_t \\
[\bigcirc\alpha]_t &\stackrel{\text{def}}{=} [\alpha]_{t+1} \\
[\alpha \,\mathcal{U}\, \beta]_t &\stackrel{\text{def}}{=} \exists x\, (t \leq x \wedge [\beta]_x \wedge \forall y\, (t \leq y < x \rightarrow [\alpha]_y)) \\
[\alpha \,\mathcal{R}\, \beta]_t &\stackrel{\text{def}}{=} \forall x\, (t \leq x \rightarrow [\beta]_x \vee \exists y\, (t \leq y < x \wedge [\alpha]_y))
\end{aligned}$$

In this tanslation, the expression $w \leq x$ stands for $w < x \vee x = w$ and the expression $[\alpha]_{t+1}$ can be seen as the abbreviation of

$$\exists y\, (t < y \wedge \neg\exists z\, (t < z \wedge z < y \wedge [\alpha]_y))\,.$$

⊠

Note how, per each atom $p \in At$ in the temporal formula φ, we get a monadic predicate $p(x)$ in the translation. The effect of this translation on the derived operators $\Diamond$ and $\Box$ yields the quite natural expressions:

$$[\Box\alpha]_t \stackrel{\text{def}}{=} \forall x\, (t \leq x \rightarrow [\alpha]_x) \qquad [\Diamond\alpha]_t \stackrel{\text{def}}{=} \exists x\, (t \leq x \wedge [\alpha]_x)$$

As a pair of examples, the translations $\Box\,(\neg p \rightarrow \bigcirc p)$ and $\Diamond p$ for $t = 0$ respectively correspond to:

$$\begin{aligned}
[\Box\,(\neg p \rightarrow \bigcirc p)]_0 &= \forall x\, (0 \leq x \ \rightarrow (\neg p(x) \rightarrow p(x+1))) \\
[\Diamond p]_0 &= \exists x (0 \leq x \wedge p(x))
\end{aligned}$$

On the other hand, every MFO($<$) formula is equivalent to an LTL one. The first proof of the equivalence between LTL and MFO($<$) , known as Kamp's

Theorem (stated below), was proven by Kamp himself in 1968 [63], although several alternative proofs of the same result [104, 43, 42, 42] are available nowadays.

Theorem 2.9 (Kamp's theorem (from [63]))**.** *For every MFO(<) formula $F(x)$ with one free variable, there is an LTL formula which is equivalent to $F(x)$ over Dedekind complete chains.* ⊠

2.3.2 Büchi automata and LTL

As we have already seen, LTL models are essentially infinite, discrete and linear sequences, with an identified start state. Thus each temporal formula corresponds to a set of models on which that formula is satisfied.

If we think of each particular state within a propositional model, then we can see that each such state is of finite size. Since there are only a finite number of propositions and a finite set of values for those propositions, we can define the set of all possible states and then rename each of them by a new symbol. Because of this representation, our models will become strings.

Considering LTL models as strings we can utilize the large amount of previous work on finite automata. In particular we can define a finite automaton that accepts *exactly* the strings we are interested in and so we can use finite automata to represent LTL models. As pioneers in this area saw, we need an specific form of automata over infinite strings, often named ω-automata [16, 119]. We next give the definition of a specific kind of ω-automaton, which is called Büchi automaton, and it is going to be the one we will use along this thesis.

Definition 2.29 (Büchi automaton)**.** *A nondeterministic Büchi automaton (BA) is a tuple $A = (Q, \Sigma, \delta, q_0, F)$ where, as happens in the case of finite automata, Q is a set of states, Σ an alphabet, δ a transition function and q_0 and F are the initial state and a set of final states respectively. The set of words with infinite length that can be formed by symbols of Σ is denoted by $\Sigma\omega$.*

A run of A on an ω-word $a_0, \cdots, a_n, \cdots \in \Sigma\omega$ consists of an infinite sequence

$$\rho = p_0 \xrightarrow{a_0} p_1 \xrightarrow{a_1} p_1 \cdots,$$

such that every $p_i \in Q$ and $p_{i+1} \in \delta(p_i, a_i)$. The set

$$inf(\rho) = \{q \in Q \mid q = p_i \text{ for infinitely many } i\text{'s}\}$$

is the set of states that occur in ρ infinitely often. A run ρ is accepted if $inf(\rho) \cap F \neq \emptyset$. A Büchi automaton accepts an ω-word $w \in \Sigma^\omega$ if it has an accepting run on w. The language recognized by a Büchi automaton consists of the set of ω-words that can be formed with symbols from Σ. ⊠

Figure 2.5 shows two Büchi automata. The automaton on the left is deterministic and accepts all ω-words over the alphabet $\{a, b\}$ that contains infinitely many occurrences of a. So for instance, it accepts a^ω, ba^ω or even $(ab)^\omega$, but not b^ω. On the other hand, the automaton on the right of the figure is not deterministic, and recognizes all ω-words over the alphabet $\{a, b\}$ that contain *finitely many* occurrences of a.

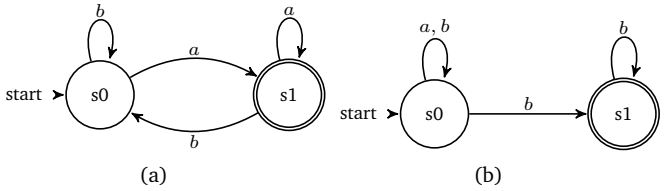


Figure 2.5: Two Büchi automata.

2.3.3 ω-languages

Büchi automata, and ω-automata in general, have been deeply studied in the literature and they are very useful tools to prove several properties of LTL such as complexity, or expressiveness. Now, we go through the ω-regular languages, the languages accepted by a Büchi automaton.

Definition 2.30 (Regular languages)**.** *Let Σ be an alphabet. The collection of regular languages over Σ, denoted by Σ^*, is defined recursively as follows:*

- *$\emptyset$ is a regular language.*
- *For each $a \in \Sigma$, the singleton language $\{a\}$ is a regular language.*
- *if $\mathcal{L}_a$ and $\mathcal{L}_b$ are regular languages then $\mathcal{L}_a \cup \mathcal{L}_b$ (union), $\mathcal{L}_a \cdot \mathcal{L}_b$ (concatenation) and $(\mathcal{L}_A)^*$ (Kleene-star) are regular.*
- *No other languages over Σ are regular.*

⊠

Definition 2.31 (ω-regular languages)**.** *Let Σ be a set of symbols (not necessarily finite), we denote by Σ^ω the set of all words of infinite length that can be built with symbols of Σ, thus an ω-language $\mathcal{L}$ over an alphabet Σ is a subset of Σ^ω. an ω-language is said to be regular it has the form:*

- *$(\mathcal{L})^\omega$ where $\mathcal{L}$ is a non-empty regular language not containing the empty string. The elements of $(\mathcal{L})^\omega$ are obtained by concatenating words from $\mathcal{L}$ infinitely many times.*
- *$\mathcal{L}_1 \cdot \mathcal{L}_2$, the concatenation of a regular language $\mathcal{L}_1$ and an ω-regular language $\mathcal{L}_2$.*
- *$\mathcal{L}_2 \cup \mathcal{L}_2$ where both $\mathcal{L}_1$ and $\mathcal{L}_2$ are ω-regular languages.*

⊠

It is known that ω-regular languages are closed under union, intersection, complementation, projection and residual. Moreover, the following relation between Büchi automata, second-order logic and ω-regular languages was proven by Büchi:

Theorem 2.10 (from [16]). *An ω-language is definable in Monadic Second Order Logic iff it is regular.* ⊠

Theorem 2.11 (from [16]). *A Büchi automaton recognizes the class of ω-regular languages.* ⊠

A subclass of ω-regular languages, which is closer to temporal logic, is the class of star-free languages. As their name suggests are constructed from the letters of the alphabet, the empty set and the operations union, concatenation, complementation but not Kleene star. Similarly to the relation between ω-regular languages, Büchi automaton and second-order logic, the theorem below shows that star-free languages are first-order definable and, hence, LTL definable.

Theorem 2.12 (from [95]). *A language is star-free iff it is first-order definable.* ⊠

This result, together with Kamp's theorem prove the relation between LTL, star-free languages and MFO($<$).

Chapter 3

Temporal Equilibrium Logic

Several approaches [10, 51] tried to provide a better nonmonotonic extensions for temporal reasoning, most of them by introducing *temporal operators* from modal temporal logics in ASP or semantics related to Logic Programming. However, these approaches impose several syntactic restrictions while their semantics are usually defined by means of reducts. Fortunately, those limitations can be overcomed by extending the definition of *Equilibrium Logic* to the temporal case. This extension, called *Temporal Equilibrium Logic*, has become the first approach to non-monotonic temporal reasoning, which does not impose any syntactic restriction.

3.1 Temporal Here-and-There

The logic of *Linear Temporal Here-and-There* (THT) is defined as follows. We start from a finite set of atoms At called the *propositional signature*. The syntax of THT is the one from propositional LTL which we recall below. A temporal formula φ is defined as:

$$\varphi ::= \bot \mid p \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \bigcirc\varphi_1 \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \mid \varphi_1 \,\mathcal{R}\, \varphi_2 \mid (\varphi_1)$$

where φ_1 and φ_2 are temporal formulae in their turn and p is any atom. A formula is said to be *non-modal* if it does not contain temporal operators. Negation is defined as $\neg\varphi \stackrel{\text{def}}{=} \varphi \rightarrow \bot$ whereas $\top \stackrel{\text{def}}{=} \neg\bot$. As usual, $\varphi \leftrightarrow \psi$ stands for $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$. Other usual temporal operators can be defined in terms of $\mathcal{U}$ and $\mathcal{R}$ as follows:

$$\Box\varphi \stackrel{\text{def}}{=} \bot \,\mathcal{R}\, \varphi \qquad \Diamond\varphi \stackrel{\text{def}}{=} \top \,\mathcal{U}\, \varphi$$

$\Box$ is read "forever" and $\Diamond$ stands for "eventually" or "at some future point." We define the following notation for a finite concatenation of $\bigcirc$'s

$$\begin{array}{lcl} \bigcirc^0\varphi & \stackrel{\text{def}}{=} & \varphi \\ \bigcirc^i\varphi & \stackrel{\text{def}}{=} & \bigcirc(\bigcirc^{i-1}\varphi) \quad \text{(with } i \geq 1\text{)} \end{array}$$

The semantics of the logic of THT is defined in terms of sequences of pairs of propositional interpretations. A (temporal) *interpretation* $\mathbf{M}$ is an infinite sequence of pairs $m_i = \langle H_i, T_i \rangle$ with $i = 0, 1, 2, \ldots$ where $H_i \subseteq T_i$ are sets of atoms standing for *here* and *there* respectively. For simplicity, given a temporal interpretation, we write $\mathbf{H}$ (resp. $\mathbf{T}$) to denote the sequence of pair components $H_0, H_1, \ldots$ (resp. $T_0, T_1, \ldots$). Using this notation, we will sometimes abbreviate the interpretation as $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$. An interpretation $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ is said to be *total* when $\mathbf{H} = \mathbf{T}$.

Definition 3.1 (THT-Satisfaction)**.** *The satisfaction relation $\models$ is interpreted as follows on THT models ($\mathbf{M}$ is a THT model and $k \in \mathbb{N}$):*

1. $\mathbf{M}, k \models p$ *iff* $p \in H_k$*, for any* $p \in At$.
2. $\mathbf{M}, k \models \varphi \wedge \psi$ *iff* $\mathbf{M}, k \models \varphi$ *and* $\mathbf{M}, k \models \psi$.
3. $\mathbf{M}, k \models \varphi \vee \psi$ *iff* $\mathbf{M}, k \models \varphi$ *or* $\mathbf{M}, k \models \psi$.
4. $\mathbf{M}, k \models \varphi \rightarrow \psi$ *iff for all* $\mathbf{H}' \in \{\mathbf{H}, \mathbf{T}\}$*,* $\langle \mathbf{H}', \mathbf{T} \rangle, k \not\models \varphi$ *or* $\langle \mathbf{H}', \mathbf{T} \rangle, k \models \psi$.
5. $\mathbf{M}, k \models \bigcirc \varphi$ *iff* $\mathbf{M}, k+1 \models \varphi$.
6. $\mathbf{M}, k \models \varphi \,\mathcal{U}\, \psi$ *iff there is* $j \geq k$ *such that* $\mathbf{M}, j \models \psi$ *and for all* $i \in [k, j-1]$*,* $\mathbf{M}, i \models \varphi$.
7. $\mathbf{M}, k \models \varphi \,\mathcal{R}\, \psi$ *iff for all* $j \geq k$ *such that* $\mathbf{M}, j \not\models \psi$*, there exists* $i \in [k, j-1]$*,* $\mathbf{M}, i \models \varphi$.
8. *never* $\mathbf{M}, k \models \bot$. ⊠

A formula φ is THT-*valid* if $\mathbf{M}, 0 \models \varphi$ for any $\mathbf{M}$. An interpretation $\mathbf{M}$ is a THT-*model* of a theory Γ, written $\mathbf{M} \models \Gamma$, if $\mathbf{M}, 0 \models \varphi$, for all formula $\varphi \in \Gamma$.

We assume that a finite sequence $\mathbf{M} = m_1, m_2, \ldots, m_n$ is an abbreviation of an infinite sequence where the remaining elements coincide with m_n, that is, for $i > n$, $m_i = m_n$. The logic of THT is an orthogonal combination of the logic of HT and the (standard) linear temporal logic (LTL). When we disregard temporal operators, we obtain the logic of HT. On the other hand, if we restrict the semantics to total interpretations, $\langle \mathbf{T}, \mathbf{T} \rangle \models \varphi$ corresponds to satisfaction of formulas $\mathbf{T} \models \varphi$ in LTL.

3.1.1 A Three-valued characterisation of THT

As we have seen in Definition 2.16, the logic of Here and There can be defined as a three valued logic. In order to extend the multivalued characterisation of HT to the temporal case, we define a valuation for any formula φ at time point i, written[1] $\mathbf{M}(i, \varphi)$, by similarly considering which formulas are satisfied by $\langle \mathbf{H}, \mathbf{T} \rangle$ (which will be assigned 2), not satisfied by $\langle \mathbf{T}, \mathbf{T} \rangle$ (which will be assigned 0) or none of the two (which will take value 1). From the definitions in the previous section, we can easily derive the following conditions:

[1] We use the same name $\mathbf{M}$ for a temporal interpretation and for its induced three-valued valuation function – ambiguity is removed by the way in which it is applied (a structure or a function on indices and formulas).

1. $\mathbf{M}(i,p) \stackrel{\text{def}}{=} \begin{cases} 2 & if \quad p \in H_i \\ 0 & if \quad p \notin T_i \\ 1 & otherwise, \quad \text{i. e. } p \in (T_i \setminus H_i) \end{cases}$

2. $\mathbf{M}(i, \varphi \wedge \psi) \stackrel{\text{def}}{=} min(\mathbf{M}(i,\varphi), \mathbf{M}(i,\psi))$

3. $\mathbf{M}(i, \varphi \vee \psi) \stackrel{\text{def}}{=} max(\mathbf{M}(i,\varphi), \mathbf{M}(i,\psi))$

4. $\mathbf{M}(i, \varphi \rightarrow \psi) \stackrel{\text{def}}{=} \begin{cases} 2 & \text{if } \mathbf{M}(i,\varphi) \leq \mathbf{M}(i,\psi) \\ \mathbf{M}(i,\psi) & \text{otherwise} \end{cases}$

5. $\mathbf{M}(i, \bigcirc\varphi) \stackrel{\text{def}}{=} \mathbf{M}(i+1, \varphi)$

6. $\mathbf{M}(i, \varphi \,\mathcal{U}\, \psi) \stackrel{\text{def}}{=} \begin{cases} 2 & \text{if} \quad \exists j \geq i : \mathbf{M}(j,\psi) = 2 \text{ and} \\ & \quad \forall k, i \leq k < j \Rightarrow \mathbf{M}(k,\varphi) = 2 \\ 0 & \text{if} \quad \forall j \geq i : \mathbf{M}(j,\psi) = 0 \text{ or} \\ & \quad \exists k, i \leq k < j, \mathbf{M}(k,\varphi) = 0 \\ 1 & \text{otherwise} \end{cases}$

7. $\mathbf{M}(i, \varphi \,\mathcal{R}\, \psi) \stackrel{\text{def}}{=} \begin{cases} 2 & \text{if} \quad \forall j \geq i : \mathbf{M}(j,\psi) = 2 \text{ or} \\ & \quad \exists k, i \leq k < j, \mathbf{M}(k,\varphi) = 2 \\ 0 & \text{if} \quad \exists j \geq i : \mathbf{M}(j,\psi) = 0 \text{ and} \\ & \quad \forall k, i \leq k < j \Rightarrow \mathbf{M}(k,\varphi) = 0 \\ 1 & \text{otherwise} \end{cases}$

From their definition, the interpretation of the temporal derived operators becomes $\mathbf{M}(i, \Box\varphi) = min\ \{\mathbf{M}(j,\varphi) \mid j \geq i\}$ and $\mathbf{M}(i, \Diamond\varphi) = max\ \{\mathbf{M}(j,\varphi) \mid j \geq 0\}$.

Under this alternative three-valued definition, an interpretation $\mathbf{M}$ *satisfies* a formula φ when $\mathbf{M}(0,\varphi) = 2$. When $\mathbf{M} = \langle \mathbf{T}, \mathbf{T} \rangle$, its induced valuation will be just written as $\mathbf{T}(i,\varphi)$ and obviously becomes a two-valued function, that is $\mathbf{T}(i,\varphi) \in \{0,2\}$.

3.1.2 Encoding THT in LTL

As we can see in Definition 3.1, the main difference with respect to LTL is the interpretation of implication (item 4) that must be checked in both components, $\mathbf{H}$ and $\mathbf{T}$, of $\mathbf{M}$. In fact, as we said before, it is easy to see that when we take total models $\mathbf{M} = \langle \mathbf{T}, \mathbf{T} \rangle$, THT satisfaction $\langle \mathbf{T}, \mathbf{T} \rangle, k \models \varphi$ collapses to standard LTL satisfaction $\mathbf{T}, k \models \varphi$ so that we will sometimes write the latter when convenient. For instance, item 4 in Definition 3.1 can be rewritten as:

4′. $\mathbf{M}, k \models \varphi \rightarrow \psi$ iff ($\mathbf{M}, k \models \varphi$ implies $\mathbf{M}, k \models \psi$) and $\mathbf{T}, k \models \varphi \rightarrow \psi$ (LTL satisfaction)

A result inherited from HT whose proof can be obtained by structural induction is the so-called *persistence* property.

Proposition 3.1 (persistence from [17]). *For any formula φ, any THT model $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ and any $i \geq 0$, if $\mathbf{M}, i \models \varphi$, then $\mathbf{T}, i \models \varphi$.* ⊠

A consequence of this proposition is that the interpretation of negation $\neg\varphi$ in $\langle \mathbf{H}, \mathbf{T}\rangle$ amounts to checking that φ does not hold in the total model $\mathbf{T}$. Formally:

Corollary 3.1. *$\langle \mathbf{H}, \mathbf{T}\rangle, i \models \neg\varphi$ iff $\mathbf{T}, i \not\models \varphi$ in LTL.* ⊠

Obviously, any THT valid formula is also LTL valid, but not the other way around. For instance, the following are THT valid equivalences:

$$\neg(\varphi \wedge \psi) \quad \leftrightarrow \quad \neg\varphi \vee \neg\psi \tag{3.1}$$
$$\neg(\varphi \vee \psi) \quad \leftrightarrow \quad \neg\varphi \wedge \neg\psi \tag{3.2}$$
$$\bigcirc(\varphi \oplus \psi) \quad \leftrightarrow \quad \bigcirc\varphi \oplus \bigcirc\psi \tag{3.3}$$
$$\bigcirc \otimes \varphi \quad \leftrightarrow \quad \otimes \bigcirc \varphi \tag{3.4}$$
$$\varphi \,\mathcal{U}\, \psi \quad \leftrightarrow \quad \psi \vee (\varphi \wedge \bigcirc(\varphi \,\mathcal{U}\, \psi)) \tag{3.5}$$
$$\varphi \,\mathcal{R}\, \psi \quad \leftrightarrow \quad \psi \wedge (\varphi \vee \bigcirc(\varphi \,\mathcal{R}\, \psi)) \tag{3.6}$$

for any binary connective $\oplus$ and any unary connective $\otimes$. Equivalences (3.1), (3.2) mean that De Morgan laws are valid whereas (3.3),(3.4) allow us to shift the '$\bigcirc$' operator to all the operands of any connective. Formulas (3.5) and (3.6) provide inductive definitions for "until" and "release" respectively. The following result captures a general class of LTL-valid formulas that are also THT-valid.

Proposition 3.2 (from [17])**.** *Let φ and ψ be two formulas not containing implication*[2]*. Then $\varphi \leftrightarrow \psi$ is THT-valid iff it is LTL-valid.* ⊠

There are, however, LTL-valid formulas that are not THT-valid. As an example, the formula $\varphi \vee \neg\varphi$ (known as *excluded middle* axiom) is not THT valid. This feature is inherited from the intermediate/intuitionistic nature of HT. In fact, the addition of this axiom makes THT collapse to LTL, much in the same way as it makes intuitionistic logic collapse to classical propositional logic. The following proposition shows that if we add a copy of this axiom for any atom at any position of the models, we can force THT models of any formula to be total.

Proposition 3.3 (from [18])**.** *Given a temporal formula φ for a propositional signature At, for every THT model $\langle \mathbf{H}, \mathbf{T}\rangle$, the propositions below are equivalent:*

(I) $\langle \mathbf{H}, \mathbf{T}\rangle, 0 \models \varphi \wedge \bigwedge_{p \in At} \Box(p \vee \neg p)$*,*

(II) $\mathbf{T}, 0 \models \varphi$ *in LTL, and for* $i \geq 0$ $H_i = T_i$

⊠

This gives us a direct way of encoding LTL in THT, since LTL models of φ coincide with its total THT models. The translation from THT to LTL can be done by extending the star-translation described in Section 2.2.3 to the temporal case by adding a translation just for modal operators:

1. $(\varphi \mathcal{U} \psi)^* \stackrel{\text{def}}{=} \varphi^* \mathcal{U} \psi^*$

[2]Remember that negation is a form of implication.

2. $(\varphi \mathcal{R} \psi)^* \stackrel{\text{def}}{=} \varphi^* \mathcal{R} \psi^*$

to the propositional case. Since $\Box$ and $\Diamond$ can be defined in terms of $\mathcal{U}$ and $\mathcal{R}$, it can be proved that $(\Box\varphi)^* = \Box\varphi^*$ and $(\Diamond\varphi)^* = \Diamond\varphi^*$.

We associate to any THT interpretation $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ the LTL interpretation $\mathbf{M}^t = \mathbf{I}$ in LTL defined as the sequence of sets of atoms $\mathbf{I} = \{I_i\}_{i \in \mathbb{N}}$ where $I_i = \{p' \mid p \in H_i\} \cup T_i$. As a THT interpretation must satisfy $H_i \subseteq T_i$ by construction, we may have LTL interpretations that do not correspond to any THT one. In particular, for an arbitrary $\mathbf{I}$, we will only be able to form some $\mathbf{M}$ such that $\mathbf{M}^t = \mathbf{I}$ when the set of primed atoms at each I_i is a subset of the non-primed ones. In other words, only LTL interpretations $\mathbf{I}$ satisfying the axiom schema:

$$\bigwedge_{p \in At} \Box(p' \rightarrow p) \tag{\textbf{Ax1}}$$

will have a corresponding THT interpretation $\mathbf{M}$ such that $\mathbf{I} = \mathbf{M}^t$.

Example 3.1. *$\mathbf{M} = ((\emptyset, \{p, q\}), (\{p\}, \{p, q\}), (\{q\}, \{q\}))$ is a THT model of the theory $\{\Box(\neg p \rightarrow q) \wedge \Diamond q\}$. In the same way, the corresponding LTL interpretation $\mathbf{M}^t = (\{p, q\}, \{p', p, q\}, \{q', q\})$ is an LTL model of*

$$\begin{array}{lcl} (\Box(\neg p \rightarrow q) \wedge \Diamond q)^* & \leftrightarrow & \Box(\neg p \rightarrow q)^* \wedge (\Diamond q)^* \\ & \leftrightarrow & \Box((\neg p \rightarrow q) \wedge ((\neg p)^* \rightarrow q')) \wedge \Diamond q' \\ & \leftrightarrow & \Box((\neg p \rightarrow q) \wedge ((\neg p \wedge \neg p') \rightarrow q')) \wedge \Diamond q'. \end{array}$$

⊠

Theorem 3.1 (from [2])**.** *Let $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ be any THT interpretation and φ any formula. For any $i \geq 0$, it holds that*

(a) *$\langle \mathbf{H}, \mathbf{T} \rangle, i \models \varphi$ if and only if $\mathbf{M}^t, i \models \varphi^*$ in LTL; and*

(b) *$\langle \mathbf{T}, \mathbf{T} \rangle, i \models \varphi$ if and only if $\mathbf{M}^t, i \models \varphi$ in LTL.* ⊠

Corollary 3.2 (from [18])**.** *Let φ be any temporal formula and let φ' be defined as $\varphi^* \wedge \bigwedge_{p \in At} \Box(p' \rightarrow p)$. The set of LTL models of φ' corresponds to the set of THT models of φ.* ⊠

3.2 Temporal Equilibrium Models

We can now proceed to describe the model selection criterion that defines temporal equilibrium models. Given two interpretations $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ and $\mathbf{M}' = \langle \mathbf{H}', \mathbf{T}' \rangle$ we say that $\mathbf{M}'$ is *lower or equal than* $\mathbf{M}$, written $\mathbf{M}' \leq \mathbf{M}$, when $\mathbf{T}' = \mathbf{T}$ and for all $i \geq 0$, $H'_i \subseteq H_i$. As usual, $\mathbf{M}' < \mathbf{M}$ stands for $\mathbf{M}' \leq \mathbf{M}$ and $\mathbf{M}' \neq \mathbf{M}$.

Definition 3.2 (temporal equilibrium model)**.** *An interpretation $\mathbf{M}$ is a* temporal equilibrium model *of a theory Γ if $\mathbf{M}$ is a total model of Γ and there is no other $\mathbf{M}' < \mathbf{M}$, such that $\mathbf{M}' \models \Gamma$.* ⊠

Note that any temporal equilibrium model is total, that is, it has the form $\langle \mathbf{T}, \mathbf{T} \rangle$ and so can be actually seen as an LTL-interpretation of the form $\mathbf{T}$ that we will call *temporal stable model*.

Definition 3.3 (temporal stable model)**.** *If $\langle \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of a theory Γ then $\mathbf{T}$ is called a* temporal stable model *of Γ (or TS-model, for short).* ⊠

In this way, since the set of LTL models of Γ coincides with the set of total THT models of Γ, we can easily see that:

Observation 3.1. *Any TS-model of a temporal theory Γ is also an LTL-model of Γ.* ⊠

Note that the consequence relation induced by temporal equilibrium models is nonmonotonic. In fact, when we restrict the syntax to ASP programs and the semantics to HT interpretations of the form $\langle H_0, T_0 \rangle$ we talk about (non-temporal) equilibrium models, which coincide with stable models in their most general definition [36]. The result below establishes a more general relation to non-temporal equilibrium logic/ASP.

Proposition 3.4 (from [23])**.** *Let Γ be a combination of non-modal connectives $\wedge, \vee, \neg, \rightarrow, \bot$, with their usual grammar, with expressions like $\bigcirc^i p$, being p an atom, and let n be the maximum value for i in all $\bigcirc^i p$ occurring in Γ. Then $\langle \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of Γ iff (1) $T_i = \emptyset$ for all $i > n$; and (2) $\langle X, X \rangle$ with $X = \bigcup_{i=0}^{n} \{\bigcirc^i p \mid p \in T_i\}$ is an equilibrium model of Γ, reading each '$\bigcirc^i p$' as a new atom in the signature.* ⊠

The TEL satisfiability problem consists in determining whether a temporal formula has a TEL model.

3.2.1 Examples

As a first example, consider the formula

$$\Box(\neg p \rightarrow \bigcirc p) \tag{3.7}$$

Its intuitive meaning corresponds to the logic program consisting of rules of the form: $p(s(X)) \leftarrow \mathit{not}\ p(X)$ where time has been reified as an extra parameter $X = 0, s(0), s(s(0)), \ldots$. Notice that the interpretation of $\neg$ is that of default negation *not* in logic programming. In this way, (3.7) is saying that, at any situation, if there is no evidence on p, then p will become true in the next state. In the initial state, we have no evidence on p, so this will imply $\bigcirc p$. To derive $\bigcirc \bigcirc p$ the only possibility would be the rule $\neg \bigcirc p \rightarrow \bigcirc \bigcirc p$, an instance of (3.7). As the body of this rule is false, $\bigcirc \bigcirc p$ becomes false by default, and so on. It is easy to see that the unique temporal stable model of (3.7) is captured by the formula $\neg p \wedge \Box(\neg p \leftrightarrow \bigcirc p)$.

As a second example, take the formula $\Diamond p$. This formula informally corresponds to an infinite disjunction $p \vee \bigcirc p \vee \bigcirc \bigcirc p \vee \ldots$. Again, as happens in disjunctive logic programming, in TEL we have a truth minimality condition that will make true the formula with as little information as possible. As a

result, it is easy to see that the temporal stable models of $\Diamond p$ are captured by the formula $\neg p \, \mathcal{U} \, (p \wedge \bigcirc\Box\neg p)$ whose models are those where p holds true at exactly one position.

It is worth noting that an LTL satisfiable formula may have no temporal stable model. As a simple example (well-known from non-temporal ASP) the logic program rule $\neg p \rightarrow p$, whose only (classical) model is $\{p\}$, has no stable models. If we assume that p cannot be derived, i.e. $\neg p$, then the rule contradicts the assumption. On the other hand, if we assume that p can be derived, then $\neg p$ becomes false and we are left with no rule that *justifies* a possible derivation for p.

When dealing with (finite) logic programs, it is well-known that non-existence of stable models is always due to a kind of cyclic dependence on default negation like this. In the temporal case, however, non-existence of temporal stable models may also be due to a lack of a finite justification for satisfying the criterion of minimal knowledge. As an example, consider the formula:

$$\Box(\neg\bigcirc p \rightarrow p) \quad \wedge \quad \Box(\bigcirc p \rightarrow p) \tag{3.8}$$

This formula has no temporal equilibrium models. To see why, note that (3.8) is LTL-equivalent (and THT-equivalent) to $\Box(\neg \bigcirc p \vee \bigcirc p \rightarrow p)$ that, in its turn, is LTL-equivalent to $\Box p$. Thus, the only LTL-model $\mathbf{T}$ of (3.8) has the form $T_i = \{p\}$ for any $i \geq 0$. However, it is easy to see that the interpretation $\langle \mathbf{H}, \mathbf{T} \rangle$ with $H_i = \emptyset$ for all $i \geq 0$ is also a THT model of (3.8), whereas $\mathbf{H} < \mathbf{T}$. It is worth to note that (3.8) was extracted from a first-order counterpart, the pair of rules $\neg p(s(X)) \rightarrow p(X)$ and $p(s(X)) \rightarrow p(X)$, that were used in [35] to show that an acyclic[3] program without a well-founded dependence ordering relation may have no stable models. In this case, we accordingly get no temporal stable models.

Another example of TEL-unsatisfiable formula is $\Box\Diamond p$, typically used in LTL to assert that property p occurs infinitely often. This formula has no temporal stable models: all models must contain infinite occurrences of p and there is no way to establish a minimal $\mathbf{H}$ among them. Thus, formula $\Box\Diamond p$ is LTL satisfiable but it has no temporal stable model. This formula is normally used in LTL to specify a *liveness* property (p occurs infinitely often). When asserted in TEL, however, this yields a conflict with the minimality criterion: informally speaking, for an infinite set of p's along time, we can always take a smaller model by removing one p. The result would also be an infinite set of p's. So, there is no way to get a minimal model. This does not mean a serious limitation in expressiveness, since for practical problems, we would usually include this type of liveness property in a constraint, rather than asserting the formula. In this way, the constraint $\neg\Box\Diamond p \rightarrow \bot$ would be ruling out all models where p does not occur intinitely often. Note also that all LTL is embeddable both in THT (Proposition 3.3) and in fact in TEL too (see [18] for details).

By contrast, the next proposition states that for a large class of temporal

[3] A logic program is *acyclic* if its corresponding dependency graph contains no cycles. This graph has as vertices the set of atoms in the program and one edge (p, q) for each rule with p in the head and q occurring in the rule body.

formulas, LTL satisfiability is equivalent to THT satisfiability and TEL satisfiability.

Proposition 3.5 (from [18]). *Let φ be temporal formula built over the connectives $\vee$, $\wedge$, $\rightarrow$, $\bigcirc$ and $\mathcal{U}$ and such that $\rightarrow$ occurs only in subformulae of the form $p \rightarrow \bot$ with $p \in At$. The propositions below are equivalent: (I) φ is LTL satisfiable; (II) φ is THT satisfiable; (III) φ has a temporal stable model, i.e. φ is TEL satisfiable.*

⊠

Theorem 3.2 (corollary 2 from [18]). *THT satisfiability problem is* PSPACE*-complete.* ⊠

3.3 Relation between THT and First-order HT

It has been shown in Section 2.3.1 that, due to Kamp's Theorem, Temporal Logic is as expressive as MFO($<$) in which only the use of one free variable is allowed, but the relation between THT and MHT($<$), the analogous version of QHT, has not been studied[4]. Our definition of MHT($<$) is characterised by the use of the set $\mathbb{N}$ of natural numbers[5] as domain as well as a set of monadic predicates with variables ranging on $\mathbb{N}$. Once the domain and the set of predicates $\mathcal{P}$ have been fixed, by $Atoms(\mathcal{P})$ we denote the set of all possible atoms formed by predicates in $\mathcal{P}$ and constants in $\mathbb{N}$. We provide next a formal definition of MHT($<$), whose corresponding interpretations are built on the set $Atoms(\mathcal{P})$.

Definition 3.4 (MHT($<$)). Monadic Quantified Here-and-There *with a linear ordering, MHT($<$), is a simplified instance of Quantified HT [101]. An MHT($<$)-*interpretation *is a tuple $\mathcal{M} = \langle \mathcal{H}, \mathcal{T} \rangle$ where $\mathcal{H} \subseteq \mathcal{T} \subseteq Atoms(\mathcal{P})$. As before, we say that the interpretation is* total *iff $\mathcal{H} = \mathcal{T}$. The satisfiability relation is defined as follows:*

- $\mathcal{M} \models p(i)$ *iff* $p(i) \in \mathcal{H}$
- $\mathcal{M} \models i \leq j$ *iff* i *is less or equal than* j *as natural numbers*
- $\wedge$, $\vee$, $\bot$ *as usual*
- $\mathcal{M} \models F \rightarrow G$ *iff for all* $w \in \{\mathcal{H}, \mathcal{T}\}$, $\langle w, \mathcal{T} \rangle \not\models F$ *or* $\langle w, \mathcal{T} \rangle \models G$
- $\mathcal{M} \models \forall x\ F(x)$ *iff for all* $i \in \mathbb{N}$, $\mathcal{M} \models F(i)$
- $\mathcal{M} \models \exists x\ F(x)$ *iff there exists* $i \in \mathbb{N}$, $\mathcal{M} \models F(i)$

⊠

[4] Up to date, the question of whether the other direction of Kamp's theorem holds for THT or not is unanswered. Namely, we ignore whether MHT($<$) can be translated back to THT.

[5] Although the original Kamp's result holds for any time model in the form of a Dedekind complete linear ordering, for our purposes, it suffices with considering the ordered set of natural numbers.

As we saw in Section 2.3.1, the trivial direction of Kamp's theorem guarantees that, given any temporal formula φ, there exists an obvious one-to-one correspondence between LTL-models of φ and MFO($<$)-models of $[\varphi]_0$. We can easily define a correspondence between THT and MHT($<$)models as follows:

Definition 3.5. *Given a THT interpretation $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ on a signature At, we say that the MHT($<$)-interpretation $\mathcal{M} = \langle \mathbb{N}, \mathcal{H}, \mathcal{T} \rangle$ corresponds to $\mathbf{M}$ iff*

- *$p \in H_i$ iff $p(i) \in \mathcal{H}$, for all $i \in \mathbb{N}$.*
- *$p \in T_i$ iff $p(i) \in \mathcal{T}$, for all $i \in \mathbb{N}$.*

⊠

We now prove that when considering this model correspondence, together with Kamp's translation defined in Section 2.3.1, we obtain the same one-to-one correspondence between THT and MHT($<$) interpretations:

Theorem 3.3. *Let φ be a temporal formula for a vocabulary Σ. Moreover, let $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ be a THT-interpretation for Σ and let $\mathcal{M} = \langle \mathcal{H}, \mathcal{T} \rangle$ be its corresponding MHT($<$)-interpretation as stated in Definition 3.5. It holds that $\mathbf{M}, i \models \varphi$ in THT iff $\mathcal{M} \models [\varphi]_i$ in MHT($<$).*

Proof. We proceed by structural induction.

- If $\varphi = \bot$ then $[\varphi]_i = \bot$ and the result is straightforward.
- If $\varphi = p$ is an atom, then $[p]_i = p(i)$ and we get the chain of equivalent conditions: $\mathbf{M}, i \models p \Leftrightarrow p \in H_i \Leftrightarrow p(i) \in \mathcal{H} \Leftrightarrow \mathcal{M} \models p(i)$.
- If $\varphi = \alpha \wedge \beta$ we get:
  $\mathbf{M}, i \models \alpha \wedge \beta$
  $\Leftrightarrow$ $\mathbf{M}, i \models \alpha$ and $\mathbf{M}, i \models \alpha$
  $\Leftrightarrow$ $\mathcal{M} \models [\alpha]_i$ and $\mathcal{M} \models [\beta]_i$ by induction on α, β
  $\Leftrightarrow$ $\mathcal{M} \models [\alpha]_i \wedge [\beta]_i$
  $\Leftrightarrow$ $\mathcal{M} \models [\alpha \wedge \beta]_i$
- The proof for $\varphi = \alpha \vee \beta$ is analogous to the one for $\alpha \wedge \beta$.
- If $\varphi = \alpha \rightarrow \beta$ we get:
  $\mathbf{M}, i \models \alpha \rightarrow \beta$
  $\Leftrightarrow$ for any $w \in \{\mathbf{H}, \mathbf{T}\}$, $\langle w, \mathbf{T} \rangle, i \not\models \alpha$ or $\langle w, \mathbf{T} \rangle, i \models \beta$

  Now, since the THT-interpretation $\langle \mathbf{T}, \mathbf{T} \rangle$ also corresponds to the MHT($<$) interpretation $\langle \mathcal{T}, \mathcal{T} \rangle$ we can apply induction on subformulas, so that we continue with the equivalent conditions:
  $\Leftrightarrow$ for any $w \in \{\mathcal{H}, \mathcal{T}\}$, $\langle w, \mathcal{T} \rangle \not\models [\alpha]_i$ or $\langle w, \mathcal{T} \rangle \models [\beta]_i$
  $\Leftrightarrow$ $\langle \mathcal{H}, \mathcal{T} \rangle \models [\alpha \rightarrow \beta]_i$.
- If $\varphi = \bigcirc\alpha$ we get the equivalent conditions:
  $\mathbf{M}, i \models \bigcirc\alpha$
  $\Leftrightarrow$ $\mathbf{M}, i+1 \models \alpha$
  $\Leftrightarrow$ $\mathcal{M} \models [\alpha]_{i+1}$ by induction
  $\Leftrightarrow$ $\mathcal{M} \models [\bigcirc\alpha]_i$

- If $\varphi = \alpha \,\mathcal{U}\, \beta$ we get the equivalent conditions:
  $\mathbf{M}, i \models \alpha \,\mathcal{U}\, \beta$
  $\Leftrightarrow$ There is some $k \geq i$ s.t. that $\mathbf{M}, k \models \beta$ and for $j \in \{i, \ldots, k-1\}$, $\mathbf{M}, j \models \alpha$

  $\Leftrightarrow$ There is some $k \geq i$ s.t. that $\mathcal{M} \models [\beta]_k$ and for $j \in \{i, \ldots, k-1\}$, $\mathcal{M} \models [\alpha]_j$

  $\Leftrightarrow$ $\mathcal{M} \models \exists k(i \leq k \wedge [\beta]_k \wedge \forall j(i \leq j < k \rightarrow [\alpha]_j))$
  $\Leftrightarrow$ $\mathcal{M} \models [\alpha \,\mathcal{U}\, \beta]_i$
- The proof for $\varphi = \alpha \,\mathcal{R}\, \beta$ is analogous to the one for $\alpha \,\mathcal{U}\, \beta$.

⊠

3.4 TEL and Infinitary Formulas

In this section we define a translation from temporal theories under Temporal Equilibrium Logic semantics into infinitary propositional formulas. Such translation is defined next:

Definition 3.6. *Let φ be a temporal formula built over a signature At. We define its expanded signature as*

$$At^{\infty} \stackrel{\text{def}}{=} \{\bigcirc^i p \mid p \in At \text{ and } i \in \mathbb{N}\}$$

We define the translation of φ into infinitary HT (HT$^{\infty}$) up to level k, written $\|\varphi\|_k$, recursively as follows:

- $\|\bot\|_k \stackrel{\text{def}}{=} \emptyset^{\vee}$
- $\|p\|_k \stackrel{\text{def}}{=} \bigcirc^k p$, *with* $p \in At$.
- $\|\bigcirc \varphi\|_k \stackrel{\text{def}}{=} \|\varphi\|_{k+1}$
- $\|\varphi \wedge \psi\|_k \stackrel{\text{def}}{=} \{\|\varphi\|_k, \|\psi\|_k\}^{\wedge}$
- $\|\varphi \vee \psi\|_k \stackrel{\text{def}}{=} \{\|\varphi\|_k, \|\psi\|_k\}^{\vee}$
- $\|\varphi \rightarrow \psi\|_k \stackrel{\text{def}}{=} \|\varphi\|_k \rightarrow \|\psi\|_k$
- $\|\Diamond\varphi\|_k \stackrel{\text{def}}{=} \{\|\varphi\|_i \mid k \leq i\}^{\vee}$
- $\|\Box\varphi\|_k \stackrel{\text{def}}{=} \{\|\varphi\|_i \mid k \leq i\}^{\wedge}$
- $\|\varphi \mathcal{U} \psi\|_k \stackrel{\text{def}}{=} \{\{\|\psi\|_i, \|\varphi\|_j \mid k \leq j < i\}^{\wedge} \mid k \leq i\}^{\vee}$
- $\|\varphi \mathcal{R} \psi\|_k \stackrel{\text{def}}{=} \{\{\|\psi\|_i, \|\varphi\|_j \mid k \leq j < i\}^{\vee} \mid k \leq i\}^{\wedge}$

⊠

We define now how a THT model is translated into HT$^{\infty}$ in the sense of [55] and then we will prove that, there exists a one-to-one correspondence between the set of THT models of a formula φ and the HT$^{\infty}$ models of $\|\varphi\|_0$.

Definition 3.7. *Let $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ be a THT interpretation. We define its corresponding HT interpretation $M^\infty = \langle H^\infty, T^\infty \rangle$ as:*

$$\begin{aligned} H^\infty &= \bigcup_{i \geq 0} \{\bigcirc^i p \mid p \in H_i\} \\ T^\infty &= \bigcup_{i \geq 0} \{\bigcirc^i p \mid p \in T_i\} \end{aligned}$$

⊠

Lemma 3.1. *Let φ be a THT formula built on a signature At, $\mathbf{M} = \langle \mathbf{H}, \mathbf{T} \rangle$ a THT interpretation and $M^\infty = \langle H^\infty, T^\infty \rangle$ its corresponding HT$^\infty$ interpretation. For all $i \in \mathbb{N}$, it holds that*

$$\mathbf{M}, i \models \varphi \Leftrightarrow M^\infty \models \|\varphi\|_i.$$

Proof. We proceed by structural induction (by (Ind.)we denote the Induction Hypothesis):

$$\begin{aligned} \mathbf{M}, i \models p, \text{ with } p \in At \quad &\Leftrightarrow \quad p \in H_i \\ &\Leftrightarrow \quad \bigcirc^i p \in H^\infty && \text{(Def. 3.7)} \\ &\Leftrightarrow \quad M^\infty \models \|p\|_i \end{aligned}$$

$$\begin{aligned} \mathbf{M}, i \models \alpha \wedge \beta \quad &\Leftrightarrow \quad \mathbf{M}, i \models \alpha \text{ and } \mathbf{M}, i \models \beta \\ &\Leftrightarrow \quad M^\infty \models \|\alpha\|_i \text{ and } M^\infty \models \|\beta\|_i && \text{(Ind.)} \\ &\Leftrightarrow \quad M^\infty \models \{\|\alpha\|_i, \|\beta\|_i\}^\wedge && \text{(Def. 2.25)} \\ &\Leftrightarrow \quad M^\infty \models \|\alpha \wedge \beta\|_i && \text{(Def. 3.6)} \end{aligned}$$

$$\begin{aligned} \mathbf{M}, i \models \alpha \vee \beta \quad &\Leftrightarrow \quad \mathbf{M}, i \models \alpha \text{ or } \mathbf{M}, i \models \beta \\ &\Leftrightarrow \quad M^\infty \models \|\alpha\|_i \text{ or } M^\infty \models \|\beta\|_i && \text{(Ind.)} \\ &\Leftrightarrow \quad M^\infty \models \{\|\alpha\|_i, \|\beta\|_i\}^\vee && \text{(Def. 2.25)} \\ &\Leftrightarrow \quad M^\infty \models \|\alpha \vee \beta\|_i && \text{(Def. 3.6)} \end{aligned}$$

$$\mathbf{M}, i \models \alpha \rightarrow \beta \quad \Leftrightarrow \quad \text{for any } w \in \{\mathbf{H}, \mathbf{T}\}, \langle w, \mathbf{T} \rangle, i \not\models \alpha \text{ or } \langle w, \mathbf{T} \rangle, i \models \beta$$

Now, since the THT-interpretation $\langle \mathbf{T}, \mathbf{T} \rangle$ also corresponds to the HT$^\infty$ interpretation $\langle T, T \rangle$, we can apply induction on subformulas, so that we continue with the equivalent conditions:

$$\begin{aligned} &\Leftrightarrow \quad \text{for any } w \in \{H^\infty, T^\infty\}, \langle w, T^\infty \rangle \not\models \|\alpha\|_i \text{ or } \langle w, T^\infty \rangle \models \|\beta\|_i && \text{(Ind.)} \\ &\Leftrightarrow \quad M^\infty \models \|\alpha \rightarrow \beta\|_i. \end{aligned}$$

$$
\begin{array}{lcll}
\mathbf{M}, i \models \alpha \mathcal{U} \beta & \Leftrightarrow & \exists j,\ i \leq j \text{ s.t. } \mathbf{M}, j \models \beta & \\
 & & \text{and } \forall k,\ \text{if } i \leq k < j \text{ then } \mathbf{M}, k \models \alpha & \\
 & \Leftrightarrow & \exists j,\ i \leq j \text{ s.t. } M^\infty, \models \|\beta\|_j & \text{(Ind.)} \\
 & & \text{and } \forall k,\ \text{if } i \leq k < j \text{ then } M^\infty \models \|\alpha\|_k & \\
 & \Leftrightarrow & M^\infty \models \{\{\|\alpha\|_k, \|\beta\|_j \mid i \leq k < j\}^\wedge \mid i \leq j\}^\vee & \text{(Def. 2.25)} \\
 & \Leftrightarrow & M^\infty \models \|\alpha \mathcal{U} \beta\|_i & \text{(Def. 3.6)}
\end{array}
$$

$$
\begin{array}{lcll}
\mathbf{M}, i \models \alpha \mathcal{R} \beta & \Leftrightarrow & \forall j,\ j \leq i \text{ if } \mathbf{M}, j \not\models \beta \text{ then} & \\
 & & \exists k,\ i \leq k < j \text{ such that } M, j \models \alpha & \\
 & \Leftrightarrow & \forall j,\ j \leq i \text{ if } M^\infty \not\models \|\beta\|_j \text{ then} & \text{(Ind.)} \\
 & & \exists k,\ i \leq k < j \text{ such that } M^\infty \models \|\alpha\|_k & \\
 & \Leftrightarrow & M^\infty \models \{\{\|\beta\|_j, \|\alpha\|_k \mid i \leq k < j\}^\vee \mid i \leq j\}^\wedge & \text{(Def. 2.25)} \\
 & \Leftrightarrow & M^\infty \models \|\alpha \mathcal{R} \beta\|_i & \text{(Def. 3.6)}
\end{array}
$$

⊠

This one-to-one correspondence together with how the minimisation criterion is defined, cause that the same correspondence remains after the minimisation. Once we have shown that Temporal Here and There can be encoded into Infinitary Formulas, we can apply all results on the latter formalism (the extension of Ferraris' definition of reduct) to the infinitary case [115].

Chapter 4

Towards an axiomatisation of THT

Usually, normal modal logics can be defined in two different ways: *semantically*, that is by describing a set of formulas that are "valid" with respect to a predefined Kripke frame, or *syntactically* (or *axiomatically*), where a logic is defined by a set of axioms and inference rules that are sufficient for deriving formulas that are "theorems".

When the set of axioms and inference rules is *finite* we are defining a *calculus*. Dealing with a calculus means that we only have at hand the axioms and inference rules; the logic represented by the calculus is the set of theorems that are deducible in it. It may happen that a given logic can be represented by different calculi, which leads to the problem of deciding whether two given axiomatic systems are equivalent (that is, both generate the same logic) or not. A formula φ is said to be *valid* for a class of frames $\mathcal{C}$ if for any kripke model $\mathcal{M}$ that can be formed with a frame from $\mathcal{C}$, $\mathcal{M}$ satisfies φ.

A desirable property of an axiomatic system is not only to be *sound* but also *complete*. Given a class of frames $\mathcal{C}$, we say that an axiomatic system $\vdash_\Lambda$ for a logic Λ is *sound* with respect to a class $\mathcal{C}$ of Kripke frames if for every formula φ of Λ it holds that

$$\text{if } \vdash_\Lambda \varphi \text{ then } \mathcal{C} \models \varphi.$$

Informally speaking, this property means that every derivable formula in the logic is also valid with respect to $\mathcal{C}$. Conversely, the axiomatic system is *complete* if for any formula φ of Λ it holds that

$$\text{if } \mathcal{C} \models \varphi \text{ then } \vdash_\Lambda \varphi,$$

or, in other words, every formula semantically valid in $\mathcal{C}$ is also derivable from the axiomatic system.

While several axiomatisations of LTL have been published in the state of the art [53, 121], Here and There was first axiomatised by Łukasiewicz [85] and later, in a slightly simplified form, by Hosoi [60]. Hosoi's axiomatic system consists of the Hilbert-type Intuitionistic Propositional calculus plus the axiom

$$\varphi \vee (\varphi \to \psi) \vee (\psi \to \bot)\,.$$

In this chapter we aim to set the bases for axiomatising THT by adapting Goldblatt's proof of completeness of LTL [53], which is based on the canonical model and filtration. We start presenting an axiomatisation of Here and There based on the concept of HT system.

4.1 HT axiomatisation

As a first building block of our THT axiomatisation, we provide a sound and complete axiomatic system for Here and There. It consists of axioms (1)-(9) and Inference Rule 10 from Table 4.1 and Modus Ponens as rule of inference. We start defining when a formula is valid in Here and There as well as when it is derivable.

Definition 4.1 (Validity)**.** *We say that a formula φ is* valid*, written $\models \varphi$, if it is a tautology in Here and There.* ⊠

Definition 4.2 (Derivability)**.** *An HT formula φ is said to be* derivable *from a set of formulas Γ, written $\Gamma \vdash \varphi$, if there exists $n \in \mathbb{N}$ and a sequence $\psi_0, \cdots, \psi_n$ of formulas such that $\psi_n = \varphi$ and for all $i \in \mathbb{N}$, if $i \leq n$ then one of the following conditions holds:*

- *ψ_i is an instance of one of the axioms* (1)-(9) *from Table 4.1,*
- *$\psi_i \in \Gamma$,*
- *ψ_i is obtained from previous formulas in the sequence by means of Modus Ponens or by Rule* (10) *from Table 4.1.*

⊠

Theorem 4.1 (Deduction Theorem adapted from [25])**.** *Let Γ be a set of formulas and both φ and ψ two formulas. It holds that*

$$\textit{if } \Gamma, \varphi \vdash_{HT} \psi \textit{ then } \Gamma \vdash_{HT} \varphi \to \psi$$

⊠

The proof of soundness consists in checking that axioms (1)-(9) shown in Table 4.1 are valid in Here and There while both (10) and Modus Ponens preserve validity. Since the only difference of our proposal with respect to Hosoi's axiomatisation is the use of (10) instead of Hosoi's axiom, we refer to [60] for the validity of HT axioms while we prove that (10) preserves validity in Lemma A.5. Concerning to completeness, we present here an alternative proof based on the concept of HT system. We begin defining the concept of *tableau*.

Table 4.1: Set of axioms and inference rules

Logic	Name	Axiom/ Inference Rule
Here and There	**Prop. Logic**	(1)$\varphi_0 \to (\varphi_1 \to \varphi_0)$ (2)$(\varphi_0 \to (\varphi_1 \to \varphi_2)) \to ((\varphi_0 \to \varphi_1) \to (\varphi_0 \to \varphi_2))$ (3)$\varphi_0 \wedge \varphi_1 \to \varphi_0$ (4)$\varphi_0 \wedge \varphi_1 \to \varphi_1$ (5)$\varphi_0 \to (\varphi_1 \to \varphi_0 \wedge \varphi_1)$ (6)$\varphi_0 \to \varphi_0 \vee \varphi_1$ (7)$\varphi_1 \to \varphi_0 \vee \varphi_1$ (8)$(\varphi_2 \to \varphi_2) \to ((\varphi_1 \to \varphi_2) \to (\varphi_0 \vee \varphi_1 \to \varphi_2))$ (9)$\bot \to \varphi_0$
		(10) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_n \to \chi_1 \vee \cdots \vee \chi_n}{\phi_1 \vee (\phi_1 \to \chi_1) \vee \cdots \vee \phi_n \vee (\phi_n \to \chi_n)}$
	Modus Ponens	(11) $\dfrac{\varphi \to \psi, \varphi}{\psi}$
K_{HT}	$\boldsymbol{K_\Box}$	(12) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \to \chi_1 \vee \cdots \vee \chi_n \vee \varphi}{\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \to \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \Box\varphi}$ (13) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \to \chi_1 \vee \cdots \vee \chi_n \vee \neg\neg\varphi}{\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \to \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \neg\neg\Box\varphi}$ (14) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \wedge \varphi \to \chi_1 \vee \cdots \vee \chi_n}{\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \wedge \Diamond\varphi \to \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n}$ (15) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \wedge \neg\neg\varphi \to \chi_1 \vee \cdots \vee \chi_n}{\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \wedge \neg\neg\Diamond\varphi \to \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n}$
	$\boldsymbol{K_\bigcirc}$	(16) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \to \chi_1 \vee \cdots \vee \chi_n \vee \varphi}{\bigcirc\phi_1 \wedge \cdots \wedge \bigcirc\phi_m \to \hat{\bigcirc}\chi_1 \vee \cdots \vee \hat{\bigcirc}\chi_n \vee \bigcirc\varphi}$ (17) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \to \chi_1 \vee \cdots \vee \chi_n \vee \neg\neg\varphi}{\bigcirc\phi_1 \wedge \cdots \wedge \bigcirc\phi_m \to \hat{\bigcirc}\chi_1 \vee \cdots \vee \hat{\bigcirc}\chi_n \vee \neg\neg \bigcirc \varphi}$ (18) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \wedge \varphi \to \chi_1 \vee \cdots \vee \chi_n}{\bigcirc\phi_1 \wedge \cdots \wedge \bigcirc\phi_m \wedge \hat{\bigcirc}\varphi \to \hat{\bigcirc}\chi_1 \vee \cdots \vee \hat{\bigcirc}\chi_n}$ (19) $\dfrac{\phi_1 \wedge \cdots \wedge \phi_m \wedge \neg\neg\varphi \to \chi_1 \vee \cdots \vee \chi_n}{\bigcirc\phi_1 \wedge \cdots \wedge \bigcirc\phi_m \wedge \neg\neg\hat{\bigcirc}\varphi \to \hat{\bigcirc}\chi_1 \vee \cdots \vee \hat{\bigcirc}\chi_n}$
	negation	(20) $\Diamond(\varphi \to \bot) \to (\Box\varphi \to \bot)$ (21) $(\Diamond\varphi \to \bot) \to \Box(\varphi \to \bot)$ (22) $\hat{\bigcirc}(\varphi \to \bot) \to (\bigcirc\varphi \to \bot)$ (23) $(\hat{\bigcirc}\varphi \to \bot) \to \bigcirc(\varphi \to \bot)$
$S4_{HT}$	**Reflexivity**	(24)$\varphi \to \Diamond\varphi$ (25)$\Box\varphi \to \varphi$
	Transitivity	(26)$\Box\varphi \to \Box\Box\varphi$ (27) $\Diamond\Diamond\varphi \to \Diamond\varphi$
THT		(28) $\Box\varphi \to \bigcirc\varphi$ (29) $\hat{\bigcirc}\varphi \to \Diamond\varphi$
	Induction	(30) $\dfrac{\varphi \to \bigcirc\varphi}{\varphi \to \Box\varphi}$ (31) $\dfrac{\hat{\bigcirc}\varphi \to \varphi}{\Diamond\varphi \to \varphi}$
	Functional	(32)$\hat{\bigcirc}\varphi \leftrightarrow \bigcirc\varphi$

4.1.1 Background on tableaux

The concept of set of formulas that can be derived from an axiomatic system is essential when proving completeness in modal and intuitionistic logics. This set of formulas can be defined in terms of a *semantic tableau* [25] (we will deal with maximal consistent tableaux), that will be used along this chapter. We provide next a formal definition of a tableau system taken from [25] together with some of its useful properties:

Definition 4.3 (Tableau from [25])**.** *A tableau consists of a pair of sets of formulas $t = (\Gamma, \Delta)$. We will say that*

- t *is* consistent *if:*

$$\forall \psi_1, \cdots, \psi_n \in \Delta \text{ then } \Gamma \nvdash \psi_1 \vee \cdots \vee \psi_n$$

  Note that the empty disjunction corresponds to $\bot$.

- t *is* maximal *if*

$$\forall \psi \ \ (\psi \in \Delta \text{ or } \psi \in \Gamma)\,.$$

- t *is* disjoint *if $\Gamma \cap \Delta = \emptyset$ and $\bot \notin \Gamma$. It should be noted that if a tableau (Γ, Δ) is both consistent and disjoint then $\bot \in \Delta$.*

- t *is* saturated *if:*
  - *If $\varphi \vee \phi \in \Gamma$ then either $\varphi \in \Gamma$ or $\phi \in \Gamma$.*
  - *If $\varphi \vee \phi \in \Delta$ then both $\varphi \in \Delta$ and $\phi \in \Delta$.*
  - *If $\varphi \wedge \phi \in \Gamma$ then both $\varphi \in \Gamma$ and $\phi \in \Gamma$.*
  - *If $\varphi \wedge \phi \in \Delta$ then either $\varphi \in \Delta$ or $\phi \in \Delta$.*
  - *If $\varphi \to \phi \in \Gamma$ then either $\varphi \in \Delta$ or $\psi \in \Gamma$*

⊠

If we focus on consistent tableaux, we will see that they are implicitly disjoint and, by Lindenbaum Lemma they can be extended to a maximal consistent one.

Lemma 4.1 (Lindenbaum Lemma from [25])**.** *Let t be a tableau. If t is consistent then it can be extended to a maximal consistent tableau t'* ⊠

Lemma 4.2. *Let $t = (\Gamma, \Delta)$ be a tableau. If t is consistent then t is disjoint.*

Proof. Suppose t is consistent but not disjoint. Hence, either $\Gamma \cap \Delta \neq \emptyset$ or $\bot \in \Gamma$. The case $\bot \in \Gamma$ would imply that t is not consistent and, therefore, a contradiction. On the other hand, if $\Gamma \cap \Delta \neq \emptyset$ then there exists a formula φ such that $\varphi \in \Gamma$ and $\varphi \in \Delta$ and, therefore, $\Gamma \vdash \varphi$. Since $\varphi \in \Delta$, t is not consistent and it contradicts our assumption. ⊠

Lemma 4.3 (Saturation)**.** *Let t be a tableau. If t is maximal and consistent then t is saturated.*

Proof. assume that t is maximal and consistent (and therefore disjoint) but it is not saturated. We consider the following cases:

- $\varphi \vee \psi \in \Gamma$ but both $\varphi \notin \Gamma$ and $\psi \notin \Gamma$: since t is maximal we get that $\varphi \in \Delta$ and $\psi \in \Delta$. As $\varphi \vee \psi \in \Gamma$ we obviously have $\Gamma \vdash \varphi \vee \psi$, which contradicts the assumption of consistency.
- $\varphi \vee \psi \in \Delta$ but either $\varphi \notin \Delta$ or $\psi \notin \Delta$: again, as t is maximal $\varphi \in \Gamma$ or $\psi \in \Gamma$. But this implies $\Gamma \vdash \varphi \vee \psi$, which is a contradiction since $\varphi \vee \psi \in \Delta$.
- $\varphi \wedge \psi \in \Gamma$ but either $\varphi \notin \Gamma$ or $\psi \notin \Gamma$: similarly to the previous cases, $\varphi \in \Delta$ or $\psi \in \Delta$ and, since $\varphi \wedge \psi \in \Gamma$, we derive the contradiction $\Gamma \vdash \varphi$ and $\Gamma \vdash \psi$.
- $\varphi \wedge \psi \in \Delta$ but both $\varphi \notin \Delta$ and $\psi \notin \Delta$: as in the previous cases, we can easily derive that $\varphi \in \Gamma$ and $\psi \in \Gamma$. Hence, from Γ, which contains both φ and ψ, we can derive $\varphi \wedge \psi$ which also belongs to Δ, contradicting the consistency criterion on t.
- $\varphi \rightarrow \psi \in \Gamma$ but both $\varphi \notin \Delta$ and $\psi \notin \Gamma$. Note that, because of the maximality and consistency critera, it holds that $\varphi \in \Gamma$ and $\psi \in \Delta$. From $\varphi \in \Gamma$ and $\varphi \rightarrow \psi \in \Gamma$ we conclude, by Modus Ponens, that $\psi \in \Gamma$, which contradicts the consistency of t.

⊠

Definition 4.4 (HT system)**.** *An* HT system $s = \langle (\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T) \rangle$*, which is a simplification of a Hintikka system for Intuitionistic logic [25], consists of a pair of maximal consistent tableaux satisfying the following conditions:*

- $\Gamma_H \subseteq \Gamma_T$ *and, as a consequence of being maximal an consistent,* $\Delta_T \subseteq \Delta_H$*;*
- *If* $\phi \rightarrow \psi \in \Delta_H$ *and* $\phi \in \Delta_H$ *then* $\phi \in \Gamma_T$ *and* $\psi \in \Delta_T$*.*

We say that s *is a system for a formula* φ *if* $\varphi \in \Delta_H$*.* ⊠

Definition 4.5 (Total System)**.** *Given an HT system* $x = \langle (\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T) \rangle$*, we define its corresponding* total system *as* $\check{x} = \langle (\Gamma_T, \Delta_T), (\Gamma_T, \Delta_T) \rangle$*.* ⊠

An interesting property of HT systems is that they can be built from a consistent tableau (by means of Lemma 4.1), as shown in the following lemma.

Lemma 4.4 (System Construction)**.** *Let* $t = (\Gamma^t, \Delta^t)$ *be a tableau. If* t *is consistent then the tableau* $t' = (\Gamma^{t'}, \Delta^{t'})$*, defined as*

$$\begin{aligned} \Gamma^{t'} &= \Gamma^t \cup \{\phi \mid \phi \rightarrow \psi \in \Delta^t \text{ and } \phi \in \Delta^t\} \\ \Delta^{t'} &= \{\psi \mid \phi \rightarrow \psi \in \Delta^t \text{ and } \phi \in \Delta^t\}, \end{aligned}$$

is also consistent.

Proof. Suppose that t is consistent but t' is not. In case that Δ^t contains no formula of the form $\phi \to \psi$ the resulting tableau $(\Gamma^t, \emptyset)$ is not consistent and, therefore, (Γ^t, Δ^t) is not consistent and we get a contradiction.
Otherwise, if we assume that Δ^t contains formulas of the form $\phi \to \psi$, then there exist m and n in $\mathbb{N}$ and formulas φ_1, ψ_1, $\cdots$, φ_{m+n}, ψ_{m+n} such that

$$\begin{aligned} \{\phi_1 \to \psi_1, \cdots, \phi_m \to \psi_m, \phi_{m+1} \to \psi_{m+1}, \cdots, \phi_{m+n} \to \psi_{m+n}\} &\subseteq \Delta^t, \\ \{\phi_1, \cdots, \phi_m, \phi_{m+1}, \cdots, \phi_{m+n}\} &\subseteq \Delta^t \end{aligned}$$

and $\Gamma_t \cup \{\phi_1, \cdots, \phi_m\} \vdash \psi_{m+1} \vee \cdots \vee \psi_{m+n}$. Therefore, we can conclude that

$$\Gamma_t \cup \{\phi_1, \cdots, \phi_m, \phi_{m+1}, \cdots, \phi_{m+n}\} \vdash \psi_1 \vee \cdots \vee \psi_m \vee \psi_{m+1} \vee \cdots \vee \psi_{m+n}. \quad (4.1)$$

By applying Lemma 4.1 on (4.1) we conclude

$$\Gamma_t \vdash \phi_1 \wedge \cdots \wedge \phi_m \wedge \phi_{m+1} \wedge \cdots \wedge \phi_{m+n} \to \psi_1 \vee \cdots \vee \psi_m \vee \psi_{m+1} \vee \cdots \vee \psi_{m+n} \quad (4.2)$$

and, therefore, the following expression can be derived by applying Rule (10) from Table 4.1:

$$\begin{aligned} \Gamma_t \vdash\ & (\phi_1 \vee (\phi_1 \to \psi_1)) \vee \cdots \vee (\phi_m \vee (\phi_m \to \psi_m)) \vee \\ & (\phi_{m+1} \vee (\phi_{m+1} \to \psi_{m+1})) \vee \cdots \vee (\phi_{m+n} \vee (\phi_{m+n} \to \phi_{m+n}))\,. \end{aligned}$$

However, all ϕ_i and $(\phi_i \to \psi_i)$, with $1 \leq i \leq m+n$, belong to Δ^t and this contradicts the consistency of t.

⊠

We introduce next some properties of HT systems we will use later, all of them related to the behaviour of connectives $\to$ and $\neg$.

Proposition 4.1. *Let $s = \langle (\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T) \rangle$ be a system. The following properties hold*

(i) Both (Γ_H, Δ_H) and (Γ_T, Δ_T) are disjoint.

(ii) Both (Γ_H, Δ_H) and (Γ_T, Δ_T) are saturated.

(iii) If $\alpha \to \beta \in \Gamma_H$ then either $\alpha \in \Delta_H$ or $\beta \in \Gamma_H$ and either $\alpha \in \Delta_T$ or $\beta \in \Gamma_T$.

(iv) If $\alpha \to \beta \in \Delta_H$ then $\alpha \in \Gamma_H$ and $\beta \in \Delta_H$ or $\alpha \in \Gamma_T$ and $\beta \in \Delta_T$.

(v) $\alpha \to \beta \in \Delta_T$ iff $\alpha \in \Gamma_T$ and $\beta \in \Delta_T$.

(vi) $\neg\neg\alpha \in \Gamma_H$ iff $\alpha \in \Gamma_T$.

⊠

4.1.2 Completeness of here and there

In this section, we use HT systems to prove the completeness of Hosoi's axiomatic system. We first define the correspondence between HT interpretations and HT systems.

Definition 4.6. *Given an HT system $s = \langle(\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T)\rangle$, let $\mathcal{M} = \langle H_s, T_s\rangle$ be its corresponding HT interpretation, which is defined as follows:*

- $H_s = \Gamma_H \cap At$ *(here, At stands for the set of propositional variables)*
- $T_s = \Gamma_T \cap At$.

Lemma 4.5 (Truth Lemma for HT)**.** *Let φ be HT formula on a set of propositional atoms At, $s = \langle(\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T)\rangle$ an HT system and both $\mathcal{M} = \langle H_s, T_s\rangle$ and $\breve{\mathcal{M}} = \langle T_s, T_s\rangle$ their corresponding HT and total-HT interpretations respectively. The following conditions hold:*

(A) If $\varphi \in \Gamma_H$ then $\mathcal{M} \models \varphi$.

(B) If $\varphi \in \Delta_H$ then $\mathcal{M} \not\models \varphi$.

(C) If $\varphi \in \Gamma_T$ then $\breve{\mathcal{M}} \models \varphi$.

(D) If $\varphi \in \Delta_T$ then $\breve{\mathcal{M}} \not\models \varphi$.

Proof. We proceed by structural induction:

Base case: $p \in At$

(A) $p \in \Gamma_H \Rightarrow p \in H_s$ (Def. 4.6)
$\Rightarrow \mathcal{M} \models p$

(B) $p \in \Delta_H \Rightarrow p \notin \Gamma_H$ (Prop. 4.1(i))
$\Rightarrow p \notin H_s$ (Def. 4.6)
$\Rightarrow \mathcal{M} \not\models p$

(C) $p \in \Gamma_T \Rightarrow p \in T_s$ (Def. 4.6)
$\Rightarrow \breve{\mathcal{M}} \models p$

(D) $p \in \Delta_T \Rightarrow p \notin \Gamma_T$ (Prop. 4.1(i))
$\Rightarrow p \notin T_s$ (Def. 4.6))
$\Rightarrow \breve{\mathcal{M}} \not\models p$

Inductive step

$$\boldsymbol{\alpha \wedge \beta}$$

(A) $\alpha \wedge \beta \in \Gamma_H \Rightarrow \alpha \in \Gamma_H$ and $\beta \in \Gamma_H$ (Prop. 4.1(ii))
$\Rightarrow \mathcal{M} \models \alpha$ and $\mathcal{M} \models \beta$ (Ind.)
$\Rightarrow \mathcal{M} \models \alpha \wedge \beta$

(B) $\alpha \wedge \beta \in \Delta_H \Rightarrow \alpha \in \Delta_H$ or $\beta \in \Delta_H$ (Prop. 4.1(ii))
$\Rightarrow \mathcal{M} \not\models \alpha$ or $\mathcal{M} \not\models \beta$ (Ind.)
$\Rightarrow \mathcal{M} \not\models \alpha \wedge \beta$

The proof for (C) and (D) would follows the same reasoning as in the cases (A) and (B) respectively.

$$\boldsymbol{\alpha \vee \beta}$$

(A) $\alpha \vee \beta \in \Gamma_H \Rightarrow \alpha \in \Gamma_H$ or $\beta \in \Gamma_H$ (Prop. 4.1(ii))
$\Rightarrow \mathcal{M} \models \alpha$ or $\mathcal{M} \models \beta$ (Ind.)
$\Rightarrow \mathcal{M} \models \alpha \vee \beta$

(B) $\alpha \vee \beta \in \Delta_H \Rightarrow \alpha \in \Delta_H$ and $\beta \in \Delta_H$ (Prop. 4.1(ii))
$\Rightarrow \mathcal{M} \not\models \alpha$ and $\mathcal{M} \not\models \beta$ (Ind.)
$\Rightarrow \mathcal{M} \not\models \alpha \vee \beta$

The proof for (C) and (D) would follows the same reasoning as in the cases (A) and (B) respectively.

$$\boldsymbol{\alpha \rightarrow \beta}$$

(A) $\alpha \rightarrow \beta \in \Gamma_H \Rightarrow \alpha \rightarrow \beta \in \Gamma_T$ ($\Gamma_H \subseteq \Gamma_T$)
$\Rightarrow \begin{cases} \alpha \in \Delta_H \text{ or } \beta \in \Gamma_H \\ and \\ \alpha \in \Delta_T \text{ or } \beta \in \Gamma_T \end{cases}$ (Prop. 4.1(iii))
$\Rightarrow \begin{cases} \mathcal{M} \not\models \alpha \text{ or } \mathcal{M} \models \beta \\ and \\ \check{\mathcal{M}} \not\models \alpha \text{ or } \check{\mathcal{M}} \models \beta \end{cases}$ (Ind.)
$\Rightarrow \mathcal{M} \models \alpha \rightarrow \beta$ (Def. 3.1)

$$\text{(B)}\quad \alpha \to \beta \in \Delta_H \quad \Rightarrow \quad \begin{cases} \alpha \in \Gamma_H \text{ and } \beta \in \Delta_H \\ \textit{or} \\ \alpha \in \Gamma_T \text{ and } \beta \in \Gamma_T \end{cases} \qquad \text{(Prop. 4.1(iv))}$$

$$\Rightarrow \quad \begin{cases} \mathcal{M} \models \alpha \text{ and } \mathcal{M} \not\models \beta \\ \textit{or} \\ \mathcal{M} \models \alpha \text{ and } \mathcal{M} \not\models \beta \end{cases} \qquad \text{(Ind.)}$$

$$\Rightarrow \quad \mathcal{M} \not\models \alpha \to \beta \qquad \text{(Def. 2.15)}$$

$$\text{(C)}\quad \alpha \to \beta \in \Gamma_T \quad \Rightarrow \quad \alpha \in \Delta_T \text{ or } \beta \in \Gamma_T \qquad \text{(Prop. 4.1(ii))}$$

$$\Rightarrow \quad \check{\mathcal{M}} \not\models \alpha \text{ or } \check{\mathcal{M}} \models \beta \qquad \text{(Ind.)}$$

$$\Rightarrow \quad \check{\mathcal{M}} \models \alpha \to \beta$$

$$\text{(D)}\quad \alpha \to \beta \in \Delta_T \quad \Rightarrow \quad \alpha \in \Gamma_T \text{ and } \beta \in \Delta_T \qquad \text{(Prop. (v))}$$

$$\Rightarrow \quad \check{\mathcal{M}} \models \alpha \text{ but } \check{\mathcal{M}} \not\models \beta \qquad \text{(Ind.)}$$

$$\Rightarrow \quad \check{\mathcal{M}} \not\models \alpha \to \beta$$

⊠

Finally, we prove that the axiomatic system is complete:

Theorem 4.2 (Completeness)**.** *Let φ be a formula. The following conditions are equivalent:*

1. $\models \varphi$
2. *There exists no system for φ.*
3. $\vdash \varphi$

Proof. We proceed as follows:

- $1 \to 2$: Suppose that $\models \varphi$ and there exists a system for φ. Hence, there exists $s = \langle (\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T) \rangle$ such that $\varphi \in \Delta_H$. Since $\varphi \in \Delta_H$, by the Lemma 4.5, $\mathcal{M}, s \not\models \varphi$, which is a contradiction.

- $2 \to 3$: Assume that there exists no system for φ and $\nvdash \varphi$. Hence, the tableau $t_0 = (\emptyset, \{\varphi\})$ is consistent. By Lemma 4.1, there exists a maximal consistent tableau $t = (\Gamma_H, \Delta_H)$ satisfying $\varphi \in \Delta_H^t$. By Lemmas 4.4 and 4.1, t can be extended to a system $s = \langle (\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T) \rangle$ such that $\varphi \in \Delta_H$. However, s a system for φ and it contradicts our initial assumption.

⊠

Once we have proven the completeness of Here and There we will try to extend this proof to THT by adapting Goldblatt's proof for LTL.

4.2 A partial axiomatisation of temporal here and there

In this section we present a partial axiomatisation of Temporal Here and There, which is based on the analogous proof for the case of LTL developed by Goldblatt [53], and also inspired in Simpson's work on *Modal Intuitionistic Logic* [110]. Our proof of completeness will use the canonical model construction (see Section 4.9) and filtration method (Section 4.2.4). However, it is still incomplete since we have neither considered the connectives $\mathcal{U}$ and $\mathcal{R}$ nor proven the corresponding Truth Lemma. We start redefining the syntax of THT, where we introduced a new operator, $\hat{\bigcirc}$, that corresponds to the dual of $\bigcirc$, and also the semantics in terms of Kripke models.

Definition 4.7 (Syntax)**.** *Given a set of propositional variables At, a THT formula φ is built from the following the grammar:*

$$\varphi := p \mid \bot \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \Box\varphi_1 \mid \Diamond\varphi_1 \mid \bigcirc\varphi_1 \mid \hat{\bigcirc}\varphi_1$$

where $p \in At$ and both φ_1 and φ_2 are THT formulas in their turn. ⊠

Definition 4.8 (Semantics)**.** *A Temporal Here and There (THT) interpretation is a Kripke structure $\mathcal{M} = \langle W, R^{\Box}, R^{\bigcirc}, H, T\rangle$ such that W is set of Kripke worlds, $R^{\Box}$ and $R^{\bigcirc}$ are two accessibility relations (the former refers to the operators $\Box$ and $\Diamond$ while the latter refers to $\bigcirc$ and $\hat{\bigcirc}$) such that:*

- *$R^{\Box}$ is reflexive and transitive.*
- *$R^{\bigcirc}$ is functional and serial.*
- *$R^{\Box}$ corresponds to the reflexive transitive closure of $R^{\bigcirc}$.*

Finally, both H and T are defined as follows:

- *$H : W \rightarrow 2^{At}$, where At is the set of atoms.*
- *$T : W \rightarrow 2^{At}$,*

satisfying

$$\forall x \in W, H(x) \subseteq T(x).$$

Using a similar notation as in in Definition 4.6, the corresponding modal HT total interpretation $\check{\mathcal{M}}$ as the structure $\langle W, R^{\Box}, R^{\bigcirc}, T, T\rangle$. The satisfacion of a formula φ is defined in a recursive way as follows:

- *$\mathcal{M}, x \not\models \bot$*
- *$\mathcal{M}, x \models p$ iff $p \in H(x)$*
- *$\mathcal{M}, x \models \varphi \vee \psi$ iff $\mathcal{M}, x \models \varphi$ or $\mathcal{M}, x \models \psi$*
- *$\mathcal{M}, x \models \varphi \wedge \psi$ iff $\mathcal{M}, x \models \varphi$ and $\mathcal{M}, x \models \psi$*

- $\mathcal{M}, x \models \varphi \to \psi$ *iff* $(\mathcal{M}, x \not\models \varphi$ *or* $\mathcal{M}, x \models \psi)$ *and* $\left(\check{\mathcal{M}}, x \not\models \varphi \text{ or } \check{\mathcal{M}}, x \models \psi\right)$
- $\mathcal{M}, x \models \Box\varphi$ *iff forall* $y \in W$ *if* $xR^{\Box}y$ *then* $\mathcal{M}, y \models \varphi$
- $\mathcal{M}, x \models \Diamond\varphi$ *iff there exists* $y \in W$ *such that* $xR^{\Box}y$ *and* $\mathcal{M}, y \models \varphi$
- $\mathcal{M}, x \models \bigcirc\varphi$ *iff forall* $y \in W$ *if* $xR^{\bigcirc}y$ *then* $\mathcal{M}, y \models \varphi$
- $\mathcal{M}, x \models \hat{\bigcirc}\varphi$ *iff there exists* $y \in W$ *such that* $xR^{\bigcirc}y$ *and* $\mathcal{M}, y \models \varphi$

⊠

4.2.1 Canonical model: definition and properties

The method of the Canonical Model has been used to prove completeness and other properties, like finite approximability, of many superintuitionistic and modal logics. The canonical model, in its former definition, consists of a Kripke structure where Kripke worlds are maximal consistent sets of formulas. In our case, those worlds will be all possible HT systems. We define the canonical model for modal HT logics below:

Definition 4.9 (Canonical Model, adapted from [25])**.** *We define the* Canonical Model *structure for our modal HT language built on a set of atoms* At*, as the structure* $\mathcal{M}_c = \langle W_c, R_c^{\Box}, R_c^{\bigcirc}, H_c, T_c\rangle$ *where:*

- W_c *is the set of all possible Here and There systems (see Definition 4.4). Note that, since* $\not\vdash_{HT} \bot$*, the tableau* $(\emptyset, \{\bot\})$ *is consistent and, because of Lemma 4.4 it can be extended to a system which belongs to* W_c*.*
- $R_c^{\Box}$ *is an accessibility relation defined among the systems in* W_c*. Given two systems,* $x = \langle(\Gamma_H^x, \Delta_H^x), (\Gamma_T^x, \Delta_T^x)\rangle$ *and* $y = \langle(\Gamma_H^y, \Delta_H^y), (\Gamma_T^y, \Delta_T^y)\rangle$*,* $x\ R_c^{\Box}\ y$ *is defined as*

$$x\ R_c^{\Box}\ y \quad \stackrel{\text{def}}{=} \quad \begin{array}{l}(\forall\varphi \text{ if } \Box\varphi \in \Gamma_H^x \text{ then } \varphi \in \Gamma_H^y)\\ \text{and}\\ (\forall\varphi \text{ if } \Diamond\varphi \in \Delta_H^x \text{ then } \varphi \in \Delta_H^y)\,.\end{array}$$

We usually refer to $x\ R_c^{\Box}y$ *by using the expression*

$$\Box\Gamma_H^x \subseteq \Gamma_H^y \text{ and } \Diamond\Delta_H^x \subseteq \Delta_H^y$$

where both sets, $\Box\Gamma$ *and* $\Diamond\Delta$*, are defined as:*

$$\begin{array}{rcl}\Box\Gamma & = & \{\varphi \mid \Box\varphi \in \Gamma\}\\ \Diamond\Delta & = & \{\varphi \mid \Diamond\varphi \in \Delta\}\end{array}$$

- *$R_c^{\bigcirc}$ is an accessibility relation that connects two systems in W_c. Given two systems, $x = \langle(\Gamma_H^x, \Delta_H^x), (\Gamma_T^x, \Delta_T^x)\rangle$ and $y = \langle(\Gamma_H^y, \Delta_H^y), (\Gamma_T^y, \Delta_T^y)\rangle$, $x\ R_c^{\bigcirc}\ y$ if*

$$x\ R_c^{\bigcirc}\ y \quad \stackrel{\text{def}}{=} \quad (\forall \varphi \text{ if } \bigcirc\varphi \in \Gamma_H^x \text{ then } \varphi \in \Gamma_H^y) \text{ and } \left(\forall \varphi \text{ if } \hat{\bigcirc}\varphi \in \Delta_H^x \text{ then } \varphi \in \Delta_H^y\right).$$

We usually refer to $x\ R_c^{\bigcirc} y$ by using the expression

$$\bigcirc\Gamma_H^x \subseteq \Gamma_H^y \text{ and } \hat{\bigcirc}\Delta_H^x \subseteq \Delta_H^y$$

where both sets, $\bigcirc\Gamma$ and $\hat{\bigcirc}\Delta$, are defined as:

$$\begin{aligned} \bigcirc\Gamma &= \{\varphi \mid \bigcirc\varphi \in \Gamma\} \\ \hat{\bigcirc}\Delta &= \{\varphi \mid \hat{\bigcirc}\varphi \in \Delta\} \end{aligned}$$

- *H_c and T_c are two evaluation functions defined as*

$$\begin{aligned} H_c : W_c \to 2^{At} &\stackrel{\text{def}}{=} H_c(\langle(\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T)\rangle) = \Gamma_H \cap At. \\ T_c : W_c \to 2^{At} &\stackrel{\text{def}}{=} T_c(\langle(\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T)\rangle) = \Gamma_T \cap At. \end{aligned}$$

Since every system $s \in W_c$ satisfies the condition $H_c(s) \subseteq T_c(s)$ then $\mathcal{M}_c$ is a model. ⊠

We present below some properties of the canonical model, which will be used along this chapter:

Lemma 4.6. *Let x and y be two systems in the canonical model $\mathcal{M}_c$. If $x\ R_c^{\Box}\ y$ then $\Box\Gamma_T^x \subseteq \Gamma_T^y$ and $\Diamond\Delta_T^x \subseteq \Delta_T^y$.*

Proof. For the sake of contradiction assume that either $\Box\Gamma_T^x \not\subseteq \Gamma_T^y$ or $\Diamond\Delta_T^x \not\subseteq \Delta_T^y$. If $\Box\Gamma_T^x \not\subseteq \Gamma_T^y$ then there exists a formula φ such that $\Box\varphi \in \Gamma_T^x$ but $\varphi \notin \Gamma_T^y$. Hence

$\Box\varphi \in \Gamma_T^x$	$\Rightarrow$	$\neg\Box\varphi \in \Delta_T^x$	($\bot \in \Delta_H^x$ and Prop. 4.1 (v))
	$\Rightarrow$	$\neg\Box\varphi \in \Delta_H^x$	($\Delta_T^x \subseteq \Delta_H^x$)
	$\Rightarrow$	$\Diamond(\varphi \to \bot) \in \Delta_H^x$	(Axiom (20) from Table 4.1)
	$\Rightarrow$	$\varphi \to \bot \in \Delta_H^y$	($xR_c^{\Box}y$)
	$\Rightarrow$	$\varphi \in \Gamma_T^y$	(Prop. 4.1(iii))

which is a contradiction. On the other hand, if $\Diamond\Delta_T^x \not\subseteq \Delta_T^y$ then there exists a formula φ such that $\Diamond\varphi \in \Delta_T^x$ but $\varphi \notin \Delta_T^y$, so we can derive

$$
\begin{array}{lcll}
\varphi \notin \Delta_T^y & \Rightarrow & \varphi \in \Gamma_T^y & \text{(Prop. 4.1(i))} \\
& \Rightarrow & \neg\varphi \in \Delta_T^y & \text{(Prop. 4.1(v))} \\
& \Rightarrow & \neg\varphi \in \Delta_H^y & (\Delta_T^y \subseteq \Delta_H^y) \\
& \Rightarrow & \Box\neg\varphi \notin \Gamma_H^x & (xR_c^{\Box}y) \\
& \Rightarrow & \Box\neg\varphi \in \Delta_H^x & \text{(Prop. 4.1(i))} \\
& \Rightarrow & \neg\Diamond\varphi \in \Delta_H^x & \text{(Axiom (21) from Table 4.1)} \\
& \Rightarrow & \Diamond\varphi \in \Gamma_T^x & \text{(Prop. 4.1(iv))}
\end{array}
$$

which contradicts the consistency of the tableau (Γ_T^x, Δ_T^x). ⊠

Lemma 4.7. *Let x and y be two systems in the canonical model $\mathcal{M}_c$. If $xR_c^{\bigcirc}y$ then $\bigcirc\Gamma_T^x \subseteq \Gamma_T^y$ and $\bigcirc\Delta_T^x \subseteq \Delta_T^y$.*

Proof. It can be done by following the same reasoning as in Lemma 4.6, but using Axioms (22) and (23) instead. ⊠

4.2.2 Canonical model: an example

As an example of a proof of completeness using the canonical model construction, we show that the axiomatic system for the logic $\mathrm{K_{HT}}$, which consists of Axioms (1)-(9), (20)-(21) from Table 4.1 and rules of inference (10)-(15) from the same table, is sound and complete with respect to all class of frames. $\mathrm{K_{HT}}$ formulas are built from the grammar

$$\varphi := p \mid \bot \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \Box\varphi_1 \mid \Diamond\varphi_1,$$

where both φ_1 and φ_2 are valid formulas in their turn, while its semantics is given in terms of Kripke structures.

Definition 4.10 ($\mathrm{K_{HT}}$ semantics)**.** *A K_{HT} interpretation is defined as the Kripke structure $\mathcal{M} = \langle W, R^{\Box}, H, T\rangle$ where W is a set of worlds, $R^{\Box}$ an accesibility relation and both H and $T : W \rightarrow 2^{At}$, with At being a signature, are two evaluation functions such that*

$$\forall x \in W, H(x) \subseteq T(x).$$

The structure $\langle W, R^{\Box}, T, T\rangle$, which is denoted by $\check{\mathcal{M}}$ is called total Kripke model*. A model $\mathcal{M}$ satisfies a formula φ in a world $x \in W$ if:*

- $\mathcal{M}, x \not\models \bot$
- $\mathcal{M}, x \models p$ *iff* $p \in H(x)$
- $\mathcal{M}, x \models \varphi \vee \psi$ *iff* $\mathcal{M}, x \models \varphi$ *or* $\mathcal{M}, x \models \psi$
- $\mathcal{M}, x \models \varphi \wedge \psi$ *iff* $\mathcal{M}, x \models \varphi$ *and* $\mathcal{M}, x \models \psi$
- $\mathcal{M}, x \models \varphi \rightarrow \psi$ *iff* $(\mathcal{M}, x \not\models \varphi$ *or* $\mathcal{M}, x \models \psi)$ *and* $\left(\check{\mathcal{M}}, x \not\models \varphi \text{ or } \check{\mathcal{M}}, x \models \psi\right)$
- $\mathcal{M}, x \models \Box\varphi$ *iff forall* $y \in W$ *if* $xR^{\Box}y$ *then* $\mathcal{M}, y \models \varphi$
- $\mathcal{M}, x \models \Diamond\varphi$ *iff there exists* $y \in W$ *such that* $xR^{\Box}y$ *and* $\mathcal{M}, y \models \varphi$

The $\mathrm{K_{HT}}$ canonical model is the structure $\mathcal{M}_c = \langle W_c, R_c^{\Box}, H_c, T_c \rangle$ where W_c, $R_c^{\Box}$ and both H_c and T_c are defined as in Definition 4.9. The following lemma, as Lemma 4.5 for HT, states that $\mathcal{M}_c$ determines the logic $\mathrm{K_{HT}}$.

Lemma 4.8 ($\mathrm{K_{HT}}$). *Let φ a $\mathrm{K_{HT}}$ formula and x a system in the canonical model $\mathcal{M}_c$. The following conditions hold:*

- *If $\varphi \in \Gamma_H^x$ then $\mathcal{M}_c, x \models \varphi$*
- *If $\varphi \in \Delta_H^x$ then $\mathcal{M}_c, x \not\models \varphi$*
- *If $\varphi \in \Gamma_T^x$ then $\check{\mathcal{M}}_c, x \models \varphi$*
- *If $\varphi \in \Delta_T^x$ then $\check{\mathcal{M}}_c, x \not\models \varphi$*

Proof. We proceed by structural induction on φ. Since the cases for atoms and connectives $\wedge$, $\vee$ and $\rightarrow$ have been proved in Lemma 4.5, we will only consider the modal connectives:

- $\varphi = \Box\alpha$:
  1. Assume that $\Box\alpha \in \Gamma_H^x$ but $\mathcal{M}_c, x \not\models \Box\alpha$. Due to the initial assumption, we assure that there exists one system $y \in W_c$ such that $xR_c^{\Box}y$ and $\mathcal{M}, y \not\models \alpha$. Finally, by the induction hypothesis we derive that $\alpha \in \Delta_H^y$ and, from $xR_c^{\Box}y$ and $\alpha \in \Delta_H^y$, $\Box\alpha \notin \Gamma_H^x$, which is a contradiction.
  2. Assume that $\Box\alpha \in \Delta_H^x$ and let $u = (\Box\Gamma_H^x, \Diamond\Delta_H^x \cup \{\alpha\})$ a consistent tableau (because of Lemma A.4). By Lemma 4.4, we can extend u to a system $y \in W_c$ such that, by definition, $xR_c^{\Box}y$ and $\alpha \in \Delta_H^y$. Hence, $\mathcal{M}_c, y \not\models \alpha$ because of the induction hypothesis and this means that $\mathcal{M}_c, x \not\models \Box\alpha$.
  3. Assume that $\Box\alpha \in \Gamma_T^x$ but $\check{\mathcal{M}}_c, x \not\models \Box\alpha$. It means that there exists a system y in W_c such that $xR_c^{\Box}y$ and $\check{\mathcal{M}}, y \not\models \alpha$. Then, by induction hypothesis, we derive $\alpha \in \Delta_T^y$. Finally, from $xR_c^{\Box}y$ and $\alpha \in \Delta_T^y$ we conclude, by Lemma 4.6, $\Box\alpha \notin \Gamma_T^x$, which is a contradiction.
  4. Since $\Box\alpha \in \Delta_T^x$, $\Box\alpha \in \Delta_H^x$. Let be $t = (\Box\Gamma_H^x, \Diamond\Delta_H^x \cup \{\neg\neg\alpha\})$ be a consistent tableau (because of Lemma A.1). By Lemma 4.4, t can be extended to a system y belonging to W_c. By definition, $xR_c^{\Box}y$ and $\neg\neg\alpha \in \Delta_H^y$, thus, $\neg\alpha \in \Gamma_T^y$ (it follows from Proposition 4.1 (iv) and $\Gamma_H^y \subseteq \Gamma_T^y$). Finally, since (Γ_T^y, Δ_T^y) is saturated, $\alpha \in \Delta_T^y$ and, by induction hypothesis, $\check{\mathcal{M}}, y \not\models \alpha$ and, therefore, $\check{\mathcal{M}}, x \not\models \Box\alpha$.
- $\varphi = \Diamond\alpha$:
  1. Assume that $\Diamond\alpha \in \Gamma_H^x$ and let $t = (\Box\Gamma_H^x \cup \{\alpha\}, \Diamond\Delta_H^x)$ be a consistent tableau (because of Lemma A.2) and $y \in W_c$ its corresponding system. By definition, $xR_c^{\Box}y$ and $\alpha \in \Gamma_H^y$. Finally applying induction on $\alpha \in \Gamma_H^y$ we get $\mathcal{M}, y \models \alpha$ and, therefore (note that $xR_c^{\Box}y$), $\mathcal{M}, x \models \Diamond\alpha$.

2. Assume that $\Diamond\alpha \in \Delta_H^x$ but $\mathcal{M}, x \models \Diamond\varphi$. Since $\mathcal{M}, x \models \Diamond\varphi$, there exists a system $y \in W_c$ such that $xR_c^{\Box}y$ and $\mathcal{M}, y \models \alpha$. By applying the induction hypothesis, we derive that $\alpha \in \Gamma_H^y$. Finally, since $xR_c^{\Box}y$ and $\Diamond\alpha \in \Delta_H^x$, we derive that $\alpha \in \Delta_H^y$, which is a contradiction.
3. Assume that $\Diamond\alpha \in \Gamma_T^x$ and let $t = (\Box\Gamma_H^x \cup \{\neg\neg\varphi\}, \Diamond\Delta_H^x)$ be a consistent tableau (because of Lemma A.3). As in the previous cases, let $y \in W_c$ be the system corresponding to u. It holds that $xR_c^{\Box}y$, $\neg\neg\alpha \in \Gamma_H^y$ and $\alpha \in \Gamma_T^y$ (because of Proposition 4.1 (vi)). Finally, by induction hypothesis, we derive $\check{\mathcal{M}}, y \models \alpha$ and, since xRy, it follows that $\check{\mathcal{M}}, x \models \Diamond\alpha$.
4. Assume that $\Diamond\alpha \in \Delta_T^x$ but $\check{\mathcal{M}}, x \models \Diamond\varphi$. Since $\check{\mathcal{M}}, x \models \Diamond\varphi$, there exists a system $y \in W_c$ such that $xR^c y$ and $\check{\mathcal{M}}, y \models \alpha$. By applying the induction hypothesis we derive that $\alpha \in \Gamma_T^y$. On the other hand, since $xR_c^{\Box}y$ and $\Diamond\alpha \in \Delta_T^x$ we derive, by Lemma 4.6, the contradiction that $\alpha \in \Delta_T^y$.

⊠

Finally use Lemma 4.8 to prove $\mathrm{K_{HT}}$ completeness.

Theorem 4.3 (Soundness and Completeness for $\mathrm{K_{HT}}$)**.**

$$\vdash_{K_{HT}} \varphi \textit{ iff } \models_{K_{HT}} \varphi$$

Proof.

- $\vdash_{K_{HT}} \varphi \Rightarrow \models_{K_{HT}} \varphi$: Rules of inference (12)-(19) preserve validity and axioms (20)-(23) are valid (see Section A.3).
- $\models_{K_{HT}} \varphi \Rightarrow \vdash_{K_{HT}} \varphi$: Suppose $\nvdash_{K_{HT}} \varphi$, then, by a similar argument used for the completeness of Here and There, there exists a system $s = \langle(\Gamma_H, \Delta_H), (\Gamma_T, \Delta_T)\rangle$ such that $\varphi \in \Delta_H$. Take now the canonical model M_c from Definition 4.9. Since $s \in W_c$ and $\varphi \in \Delta_H$ then by Truth Lemma for $\mathrm{K_{HT}}$ (Lemma 4.8), $\mathcal{M}_c, s \not\models_{K_{HT}} \varphi$ which contradicts the assumption $\models_{K_{HT}} \varphi$.

⊠

4.2.3 Back to THT, properties of $R_c^{\Box}$ and $R_c^{\bigcirc}$

THT frames are characterised by some conditions imposed on $R_c^{\Box}$ and $R_c^{\bigcirc}$: while $R_c^{\Box}$ is reflexive and transitive, $R_c^{\bigcirc}$ is serial and functional. The following lemmas show that both relations satisfy these properties:

Lemma 4.9 ($R_c^{\Box}$ is reflexive and transitive)**.** *The set of axioms* (24)-(27) *from Table 4.1 guarantees that $R_c^{\Box}$ is both reflexive and transitive.*

Proof. We will proceed by contradiction in all cases. If we suppose that $R_c^{\Box}$ is not reflexive, then there exists a system $x \in W_c$ such that $x \not R_c^{\Box} x$ thus there exists a formula φ such that either $\Box\varphi \in \Gamma_H^x$ but $\varphi \notin \Gamma_H^x$ or $\Diamond\varphi \in \Delta_H^x$ and $\varphi \notin \Delta_H^x$. In the first case we get

$$\Box\varphi \in \Gamma_H^x \quad \Rightarrow \quad \varphi \in \Gamma_H^x \qquad \text{(Axiom (25) from Table 4.1)}$$

which is a contradiction. In the second case we have

$$\begin{array}{lll} \varphi \notin \Delta_H^x & \Rightarrow & \varphi \in \Gamma_H^x \qquad \text{(Prop. 4.1(i))} \\ & \Rightarrow & \Diamond\varphi \in \Gamma_H^x \qquad \text{(Axiom (24) from Table 4.1)} \end{array}$$

which is also a contradiction.

If we assume that $R_c^{\Box}$ is not transitive then, there exist $x, y, z \in W_c$ such that $xR_c^{\Box}y$, $yR_c^{\Box}z$ but $x\not R_c^{\Box}z$. From $x\not R_c^{\Box}\ z$ we can conclude that there exists a formula φ such that either $\Box\varphi \in \Gamma_H^x$ and $\varphi \notin \Gamma_H^z$ or $\Diamond\varphi \in \Delta_H^x$ and $\varphi \notin \Delta_H^z$. In the first case we get

$$\begin{array}{lll} \Box\varphi \in \Gamma_H^x & \Rightarrow & \Box\Box\varphi \in \Gamma_H^x \qquad \text{(Axiom (26) from Table 4.1)} \\ & \Rightarrow & \Box\varphi \in \Gamma_H^y \qquad (xR_c^{\Box}y) \\ & \Rightarrow & \varphi \in \Gamma_H^z \qquad (yR_c^{\Box}z) \end{array}$$

which contradicts the consistency of the tableau (Γ_H^z, Δ_H^z). In the second case we proceed as follows:

$$\begin{array}{lll} \Diamond\varphi \in \Delta_H^x & \Rightarrow & \Diamond\Diamond\varphi \in \Delta_H^x \qquad \text{(Axiom (27) from Table 4.1))} \\ & \Rightarrow & \Diamond\varphi \in \Delta_H^y \qquad (x\ R_c^{\Box}\ y) \\ & \Rightarrow & \varphi \in \Delta_H^z \qquad (y\ R_c^{\Box}\ z) \end{array}$$

which is also a contradiction ⊠

Lemma 4.10 (Seriality of $R_c^{\bigcirc}$)**.** *$R_c^{\bigcirc}$ is serial, that is, it satisfies the following property:*

$$\forall x \in W_c, \exists y \in W_c \textit{ s.t. } xR_c^{\bigcirc}y$$

Proof. Let $x \in W_c$ be a system and φ a formula. Since Axiom (32) from Table 4.1 belongs to Γ_H and (Γ_H^x, Δ_H^x) is saturated, we get that either both $\bigcirc\varphi$ and $\hat{\bigcirc}\varphi$ belong to Γ_H^x or both belong to Δ_H^x. In the first case we can define the tableau

$$(\bigcirc\Gamma_H^x \cup \{\varphi\}, \hat{\bigcirc}\Delta_H^x)$$

which, because of Corollary A.1, is consistent and therefore it can be extended to a system y such that $y \in W_c$ and, by definition, $xR_c^{\bigcirc}y$. In the second case, the tableau

$$(\bigcirc\Gamma_H^x, \hat{\bigcirc}\Delta_H^x \cup \{\varphi\}),$$

is consistent too (Corollary A.2), so it can also be extended to a system y such that $xR_c^{\bigcirc}y$. In both cases we have found a system y which is accessible from x by means of $R_c^{\bigcirc}$ ⊠

Lemma 4.11 ($R_c^{\bigcirc}$ is functional)**.** *The relation $R_c^{\bigcirc}$ is functional, that is, it satisfies the expression:*

$$\forall x,\ y,\ z \in W_c,\ \textit{if } xR_c^{\bigcirc}y \textit{ and } xR_c^{\bigcirc}z \textit{ then } y = z$$

Proof. Assume that $R_c^{\bigcirc}$ is not functional, so we get that

$$\exists x,\ y,\ z \in W_c \text{ s.t } xR_c^{\bigcirc}y \text{ and } xR_c^{\bigcirc}z \text{ but } y \neq z.$$

Since $y \neq z$ we must consider two different cases:

$$\begin{array}{lcll} \varphi \in \Gamma_H^y \text{ and } \varphi \in \Delta_H^z & \Leftrightarrow & \hat{\bigcirc}\varphi \notin \Delta_H^x \text{ and } \bigcirc\varphi \notin \Gamma_H^x & (xR_c^{\bigcirc}y \text{ and } xR_c^{\bigcirc}z) \\ & \Leftrightarrow & \hat{\bigcirc}\varphi \in \Gamma_H^x \text{ and } \bigcirc\varphi \in \Delta_H^x & \text{(Prop. 4.1(i))} \end{array}$$

which contradicts the Axiom (32) of Table 4.1. The proof for the case $\varphi \in \Gamma_T^y$ and $\varphi \in \Delta_T^z$ would be performed by using Lemma 4.6 and the Axiom (32) of Table 4.1. ⊠

4.2.4 Ancestral lemma and filtration

Apart from the conditions imposed in the previous section, we need that

$$\left(R_c^{\bigcirc}\right)^* = R_c^{\Box},$$

that is, $R_c^{\Box}$ has to be equal to the reflexive transitive closure of $R_c^{\bigcirc}$. Unfortunately it cannot be proved unless we use the filtration method. The filtration technique consists in, given a formula φ and a Kripke model $\mathcal{M}$ (like, for instance, the canonical model), identify kripke points of $\mathcal{M}$ with same truth values for φ and its subformulas and collapse them into equivalence clases. This method, which may succeed even if the canonical model fails in proving completeness, is formally defined next:

Definition 4.11. *A set Σ is* closed under subformulas *if, for all formula $\varphi \in \Sigma$, all subformulas of φ belong to Σ.* ⊠

Since we are interested only in the truth values of φ, which comes from the truth values of its subformulas, filtration is defined with respect to a set closed under subformulas. Filtration is usually defined with respect to an arbitrary Kripke model but, since our structures contain two accesibility relations, $R^{\Box}$ and $R^{\bigcirc}$, we define filtration considering this specific kind of structures. Our definition also differs from the traditional one [53] in that ours is weaker but sufficient to adapt Goldblatt's argument to THT.

Definition 4.12 (Filtration)**.** *Let Σ be a THT set closed under subformulas and $\mathcal{M} = \langle W, R^{\Box}, R^{\bigcirc}, H, T\rangle$ a modal HT model. We define the equivalence relation $\sim_\Sigma$ on W as:*

$$\begin{array}{lll} x \sim_\Sigma y & \textit{iff} & \forall\varphi \in \Sigma\,(\mathcal{M}, x \models \varphi \Leftrightarrow \mathcal{M}, y \models \varphi) \text{ and } \left(\check{\mathcal{M}}, x \models \varphi \Leftrightarrow \check{\mathcal{M}}, y \models \varphi\right) \\ & \textit{iff} & |x| = |y|. \end{array}$$

The resulting Kripke structure $\mathcal{M}_\Sigma = \langle W_\Sigma, R_\Sigma^{\Box}, R_\Sigma^{\bigcirc}, H_\Sigma, T_\Sigma\rangle$, where $W_\Sigma = W\backslash\sim_\Sigma$, is said to be a filtration of $\mathcal{M}$ with respect to Σ iff it satisfies the following conditions:

(i) For all $x, y \in W$, if $xR^{\Box}y$, then there exists $z \in W$ such that $|x|R^{\Box}_{\Sigma}|z|$ and either $|z| = |y|$ or $|z| = |\check{y}|$, where y and $\check{y}$ are a HT Kripke world and its corresponding total world (see Definition 4.5) respectively.

(ii) For all $x, y \in W$, if $x\ R^{\Box}y$, then there exists $z \in W$ such that $|z|R^{\Box}_{\Sigma}|y|$ and either $|z| = |x|$ or $|z| = |\check{x}|$, where x and $\check{x}$ are a HT Kripke world and its corresponding total world (see Definition 4.5) respectively.

(iii) For all $x, y \in W$, if $|x|R^{\Box}_{\Sigma}|y|$, then for all formulas φ, if $\Box\varphi \in \Sigma$ and $\mathcal{M}, x \models \Box\varphi$ then $\mathcal{M}, y \models \varphi$.

(iv) For all $x, y \in W$, if $|x|R^{\Box}_{\Sigma}|y|$, then for all formulas φ, if $\Diamond\varphi \in \Sigma$ and $\mathcal{M}, x \not\models \Diamond\varphi$ then $\mathcal{M}, y \not\models \varphi$.

(v) For all $x, y \in W$, if $xR^{\bigcirc}y$, then there exists $z \in W$ such that $|x|R^{\bigcirc}_{\Sigma}|z|$ and either $|z| = |y|$ or $|z| = |\check{y}|$.

(vi) For all $x, y \in W$, if $xR^{\bigcirc}y$, then there exists $z \in W$ such that $|z|R^{\bigcirc}_{\Sigma}|y|$ and either $|z| = |x|$ or $|z| = |\check{x}|$.

(vii) For all $x, y \in W$, if $|x|R^{\bigcirc}_{\Sigma}|y|$, then for all formulas φ, if $\bigcirc\varphi \in \Sigma$ and $\mathcal{M}, x \models \bigcirc\varphi$ then $\mathcal{M}, y \models \varphi$.

(viii) For all $x, y \in W$, if $|x|R^{\bigcirc}_{\Sigma}|y|$, then for all formulas φ, if $\hat{\bigcirc}\varphi \in \Sigma$ and $\mathcal{M}, x \not\models \hat{\bigcirc}\varphi$ then $\mathcal{M}, y \not\models \varphi$.

(ix) For all atoms p, if $p \in \Sigma$ then for all $x \in W$, $|x| \in H_{\Sigma}(p)$ iff $x \in H(p)$.

(x) For all atoms p, if $p \in \Sigma$ then for all $x \in W$, $|x| \in T_{\Sigma}(p)$ iff $x \in T(p)$.

⊠

Definition 4.13. *Now let $\mathcal{M}_c = \langle W_c, R^{\Box}_c, R^{\bigcirc}_c, H_c, T_c\rangle$ be the canonical model from Definition 4.9 and Σ a theory closed under subformulas . We define its corresponding filtration model as follows:*

$$\mathcal{M}_{\Sigma} = \langle W_{\Sigma}, R^{\Box}_{\Sigma}, R^{\bigcirc}_{\Sigma}, H_{\Sigma}, T_{\Sigma}\rangle$$

where

- $W_{\Sigma} = W_c \backslash \sim_{\Sigma}$.
- $|x|R^{\bigcirc}_{\Sigma}|y|$ *iff there exists* $z, t \in W_c$ *such that* $x \sim_{\Sigma} z$, $y \sim_{\Sigma} t$ *and* $z\ R^{\bigcirc}_c\ t$.
- $|x|R^{\Box}_{\Sigma}|y|$ *iff there exists* $n \in \mathbb{N}$ *such that* $|x| \left(R^{\bigcirc}_{\Sigma}\right)^n |y|$.
- $|x| \in H_{\Sigma}(p)$ *iff* $x \in H_c(p)$ *for every atom* $p \in \Sigma$.
- $|x| \in T_{\Sigma}(p)$ *iff* $x \in T_c(p)$ *for every atom* $p \in \Sigma$.

⊠

We now proceed to check that $\mathcal{M}_{\Sigma}$ is a filtration of $\mathcal{M}_c$. Note that the equivalence relation $\sim_{\Sigma}$ already satisfies conditions (ix) and (x) of Definition 4.12. We start checking that $R^{\bigcirc}_{\Sigma}$ satisfies conditions (v)-(viii) of Definition 4.12.

Lemma 4.12. *$R^{\bigcirc}_{\Sigma}$ satisfies conditions (v)-(viii) of Definition 4.12.*

Proof.

- **Condition (v)**: assume that $R^{\bigcirc}_{\Sigma}$ does not satisfy condition (v). This means that there exist two points $x, y \in W_c$ such that $xR^{\bigcirc}_c y$ but for all $z \in W_c$, if $|x|R^{\bigcirc}_{\Sigma}|z|$ then $|z| \neq |y|$ and $|z| \neq |\check{y}|$, but taking $z = y$, as $xR^{\bigcirc}_c y$ we have $|x|R^{\bigcirc}_{\Sigma}|y|$ and the contradiction $|y| \neq |y|$.

- **Condition (vi)**: Assume that $R^{\bigcirc}_{\Sigma}$ does not satisfy the condition (vi). It would mean that there exist two points $x, y \in \Sigma$ such that $xR^{\bigcirc}_c y$ but for all $z \in W_c$ such that $|z|R^{\bigcirc}_{\Sigma}|y|$ then $|z| \neq |x|$ and $|z| \neq |\check{x}|$. However, if we take $z = x$, it holds that $|x|R^{\bigcirc}_{\Sigma}|y|$ and therefore we can derive the contradiction $|x| \neq |x|$.

- **Condition (vii)**: Assume that there exist $x, y \in W_c$ and a formula $\varphi \in \Sigma$ such that $|x|R^{\bigcirc}_{\Sigma}|y|$ and $\mathcal{M}_c, x \models \bigcirc\varphi$ but $\mathcal{M}_c, y \not\models \varphi$. From the definition

$$|x|R^{\bigcirc}_{\Sigma}|y| \Leftrightarrow \exists x' \in |x|, y' \in |y| \text{ and } x' R^{\bigcirc}_c y'.$$

  and the assumptions $\mathcal{M}, x \models \bigcirc\varphi$ and $x \sim_{\Sigma} x'$ we conclude both $\mathcal{M}, x' \models \bigcirc\varphi$ and $\mathcal{M}, y' \models \varphi$. Finally, since $\mathcal{M}, y' \models \varphi$ and $y \sim_{\Sigma} y'$ we derive that $\mathcal{M}, y \models \varphi$, which is a contradiction.

- **Condition (viii)**: Assume that there exist x and y in W_c and a formula $\varphi \in \Sigma$ such that $|x|R^{\bigcirc}_{\Sigma}|y|$ and $\mathcal{M}_c, x \not\models \hat{\bigcirc}\varphi$ but $\mathcal{M}_c, y \models \varphi$. From the definition

$$|x|R^{\bigcirc}_{\Sigma}|y| \stackrel{\text{def}}{=} \exists x' \in |x|, y' \in |y| \text{ and } x' R^{\bigcirc}_c y'$$

  and the facts $y \sim_{\Sigma} y'$ we conclude that $\mathcal{M}, y' \models \varphi$ and therefore $\mathcal{M}_c, x' \models \hat{\bigcirc}\varphi$. Finally, since $x' \sim_{\Sigma} x$ then we derive that $\mathcal{M}_c, x \models \hat{\bigcirc}\varphi$, which is a contradiction.

⊠

Next step is to prove that $R^{\Box}_{\Sigma}$ also satisfies (i)-(iv) of Definition 4.12, but this step requires to introduce the following auxiliary lemma:

Lemma 4.13 (Definability Lemma adapted from [53])**.** *For any set $X \subseteq W_{\Sigma}$ there is a formula Ψ_X (a truth-functional combination of members of Σ) such that for all $x \in W_c$*

$$\mathcal{M}_c, x \models \Psi_X \textit{ iff } \exists |z| \in X \textit{ s. t. } |x| = |z| \textit{ or } |x| = |\check{z}|.$$

Proof. For each $t = \langle (\Gamma^t_H, \Delta^t_H), (\Gamma^t_T, \Delta^t_T) \rangle$, let Φ_t be the following conjunction of formulas (for a better readability we assume that both φ and ψ belong to Σ):

$$\Phi_t = \left(\bigwedge_{\varphi \in \Gamma_H^t} \varphi \right) \wedge \left(\bigwedge_{\varphi \in \Delta_T^t} \neg\varphi \right) \wedge \left(\bigwedge_{\varphi \in \left(\Gamma_T^t \bigcap \Delta_H^t\right)} \neg\neg\varphi \right) \wedge \left(\bigwedge_{\varphi,\psi \in \left(\Gamma_T^t \bigcap \Delta_H^t\right)} (\varphi \to \psi) \right).$$

By using a similar argument as in Proposition 2.5, we deduce that for all $s \in W_c$

$$\mathcal{M}_c, s \models \Phi_t \text{ iff } |s| = |t| \text{ or } |s| = |\check{t}|,$$

where $\check{t} = \langle (\Gamma_T^t, \Delta_T^t), (\Gamma_T^t, \Delta_T^t) \rangle$ (see Definition 4.5). Now W_Σ is finite, since Σ is. So if

$$X = \{|t_1|, \cdots, |t_n|\}$$

we can take

$$\Psi_X = \bigvee_{|t_i| \in X} \Phi_{t_i}.$$

⊠

Now we can prove that $R_\Sigma^\Box$ satisfies conditions (i)-(iv):

Lemma 4.14. *$R_\Sigma^\Box$ satisfies conditions (i)-(iv) of Definition 4.12*

Proof.

- **Condition (i)**: Suppose that there exists $x, y \in W_c$ such that $xR_c^\Box y$ and take

  $$X = \{|z| \text{ s. t. } \exists t \in W_c \; |x| R_\Sigma^\Box |t| \text{ and either } |t| = |z| \text{ or } |t| = |\check{z}|\}.$$

  Its complementary set is denoted by $(W_\Sigma \setminus X)$. Note that, because of how X is defined, $|x| \in X$ and, moreover, it is easy to prove that

  $$\text{if } |z| \in (W_\Sigma \setminus X) \text{ then } |\check{z}| \in (W_\Sigma \setminus X). \tag{4.3}$$

  From Lemma 4.13 (Definability Lemma) and (4.3) we conclude that for all $u \in W_c$

  $$\mathcal{M}_c, u \models \Psi_{(W_\Sigma \setminus X)} \text{ iff } \exists\, |z| \in (W_\Sigma \setminus X) \text{ s.t. } |u| = |z| \text{ or } |u| = |\check{z}| \tag{4.4}$$
  $$\text{iff } |u| \in (W_\Sigma \setminus X) \tag{4.5}$$

  We aim to show that $|y| \in X$ so, suppose by contradiction that $|y| \in (W_\Sigma \setminus X)$. Therefore, from (4.5), we conclude that $\mathcal{M}_c, y \models \Psi_{(W_\Sigma \setminus X)}$ and due to the fact $xR_c^\Box y$ we obtain $\mathcal{M}_c, x \models \Diamond \Psi_{(W_\Sigma \setminus X)}$.

On the other hand, as $|x| \in X$, we derive that $|x| \notin (W_\Sigma \setminus X)$ and, because of (4.5), we obtain $\mathcal{M}_c, x \not\models \Psi_{(W_\Sigma \setminus X)}$. Hence

$$\mathcal{M}_c, x \not\models \Diamond\Psi_{(W_\Sigma \setminus X)} \rightarrow \Psi_{(W_\Sigma \setminus X)}$$

and, consequently, it follows that

$$\not\vdash \Diamond\Psi_{(W_\Sigma \setminus X)} \rightarrow \Psi_{(W_\Sigma \setminus X)}$$

and, due to Rule (31) from Table 4.1, we get

$$\not\vdash \hat{\bigcirc}\Psi_{(W_\Sigma \setminus X)} \rightarrow \Psi_{(W_\Sigma \setminus X)},$$

which means that

$$\exists u \in W_c \text{ s.t. } \mathcal{M}_c, u \models \hat{\bigcirc}\Psi_{(W_\Sigma \setminus X)} \text{ but } \mathcal{M}_c, u \not\models \Psi_{(W_\Sigma \setminus X)}.$$

While from $\mathcal{M}_c, u \not\models \Psi_{(W_\Sigma \setminus X)}$ and (4.4) we conclude that $|u| \in X$ which, by definition of X, means that

$$\exists t \in W_c \text{ s.t. } |x| R_\Sigma^{\Box} |t| \text{ and either } |t| = |u| \text{ or } |t| = |\check{u}|, \tag{4.6}$$

from $\mathcal{M}_c, u \models \hat{\bigcirc}\Psi_{(W_\Sigma \setminus X)}$ we get the expression

$$\exists v \in W_c \text{ s.t. } u R_c^{\bigcirc} v \text{ and } \mathcal{M}_c, v \models \Psi_{(W_\Sigma \setminus X)}. \tag{4.7}$$

This means that $|v| \in (W_\Sigma \setminus X)$ and, therefore, $|\check{v}| \in (W_\Sigma \setminus X)$. We consider now the two cases in (4.6).

- $|t| = |u|$: from $u R_c^{\bigcirc} v$, $|x| R_\Sigma^{\Box} |u|$ and the Condition (v) of filtration, we derive

  $$\exists |z| \text{ s. t. } |x| R_\Sigma^{\Box} |z| \text{ and } |z| = |v| \text{ or } |z| = |\check{v}|.$$

  From this, it follows the contradiction $|v| \in X$.
- $|t| = |\check{u}|$: $\check{u} R_c^{\bigcirc} \check{v}$ follows from $u R_c^{\bigcirc} v$ and Proposition A.1. Moreover, Condition (v) allows us to derive $|\check{u}| R_\Sigma^{\bigcirc} |\check{v}|$, which together with $|x| R_\Sigma^{\Box} |\check{u}|$, leads to $|x| R_\Sigma^{\Box} |\check{v}|$. This fact means that $\check{v} \in X$, but it is a contradiction.

Summing up, it follows that our, $|y| \in (W_\Sigma \setminus X)$, leads to a contradiction.

- **Condition (ii)**: suppose that there exists $x, y \in W_c$ such that $x R_c^{\Box} y$ and let

$$Y = \{|z| \text{ s. t. } \exists t \in W_c \; |t| R_\Sigma^{\Box} |y| \text{ and either } |t| = |z| \text{ or } |t| = |\check{z}|\}$$

be a set whose complementary is denoted by $(W_\Sigma \setminus Y)$. As in Condition (i), it holds that $|y| \in Y$ and, moreover, it can be easily proved that

$$\text{if } |z| \in (W_\Sigma \setminus Y) \text{ then } |\check{z}| \in (W_\Sigma \setminus Y)\,. \tag{4.8}$$

Following the same argument as in Condition (i), it follows that for all $u \in W_c$

$$\mathcal{M}_c, u \models \Psi_{(W_\Sigma \setminus Y)} \text{ iff } \exists\, |z| \in (W_\Sigma \setminus Y) \text{ s.t. } |u| = |z| \text{ or } |u| = |\check{z}| \tag{4.9}$$
$$\text{iff} \quad |u| \in (W_\Sigma \setminus Y)\ . \tag{4.10}$$

Since what we aim is to show that $|x| \in Y$, suppose by contradiction that $|x| \in (W_\Sigma \setminus Y)$. From $|y| \in Y$ and (4.10) we obtain that $\mathcal{M}_c, y \not\models \Psi_{(W_\Sigma \setminus Y)}$ so $\mathcal{M}_c, x \not\models \Box\Psi_{(W_\Sigma \setminus Y)}$ (because $xR_c^{\Box}y$).

On the other hand, from our initial assumption, $|x| \in (W_\Sigma \setminus Y)$, we get that $\mathcal{M}_c, x \models \Psi_{(W_\Sigma \setminus Y)}$ and, therefore, $\mathcal{M}_c, x \not\models \Psi_{(W_\Sigma \setminus Y)} \rightarrow \Box\Psi_{(W_\Sigma \setminus Y)}$. This implies that $\not\vdash \Psi_{(W_\Sigma \setminus Y)} \rightarrow \Box\Psi_{(W_\Sigma \setminus Y)}$, which together with Rule (31) from Table 4.1 allows us to conclude $\not\vdash \Psi_{(W_\Sigma \setminus Y)} \rightarrow \bigcirc\Psi_{(W_\Sigma \setminus Y)}$, which means that

$$\exists u \in W_c \text{ s.t. } \mathcal{M}_c, u \models \Psi_{(W_\Sigma \setminus Y)} \text{ but } \mathcal{M}_c, u \not\models \bigcirc\Psi_{(W_\Sigma \setminus Y)}.$$

While $\mathcal{M}_c, u \models \Psi_{(W_\Sigma \setminus Y)}$ and (4.10) allow us to derive $|u| \in (W_\Sigma \setminus Y)$, from $\mathcal{M}_c, u \not\models \bigcirc\Psi_{(W_\Sigma \setminus Y)}$ we have that

$$\exists v \in W_c \text{ s.t. } uR_c^{\bigcirc}v \text{ and } \mathcal{M}_c, v \not\models \Psi_{(W_\Sigma \setminus Y)},$$

which, because of the expression (4.10), means that $|v| \in Y$, which corresponds to the complementary set of $(W_\Sigma \setminus Y)$. Since $|v| \in Y$, we use the definition of Y to conclude

$$\exists t \in W_c \text{ s.t. } |t| R_\Sigma^{\Box} |y| \text{ and } |t| = |v| \text{ or } |t| = |\check{v}|, \tag{4.11}$$

and we go now through (4.11) by cases:

- $|t| = |v|$: we get that $|v| R_\Sigma^{\Box} |y|$, which together with $uR_c^{\bigcirc}v$ and Condition (vi) of filtration, lead to the expression

  $$\exists z \in W_c \text{ s. t. } |z| R_\Sigma^{\Box} |y| \text{ and } |z| = |u| \text{ or } |z| = |\check{u}|.$$

  This would mean, by definition of Y, that $|u| \in Y$, which is a contradiction.
- $|t| = |\check{v}|$: from $uR_c^{\bigcirc}v$ and Proposition A.1 we conclude that $\check{u}R_c^{\bigcirc}\check{v}$ and, following the same reasoning as in the previous case, we get $|\check{u}| R_\Sigma^{\Box} |y|$, which contradicts the fact that both $|u|$ and $|\check{u}|$ belong to $(W_\Sigma \setminus Y)$.

From all the previous derivations, it follows that our assumption $|x| \in (W_\Sigma \setminus Y)$ leads to a contradiction.

- **Condition (iii)**: Let $\Box\varphi \in \Sigma$ be a formula and $x \in W_c$ satisfying $\mathcal{M}_c, x \models \Box\varphi$. Remark that $R^{\Box}_{\Sigma}$, which is defined as

$$|x| R^{\Box}_{\Sigma} |y| \Leftrightarrow \exists n \in \mathbb{N} \text{ s.t. } |x| \left(R^{\bigcirc}_{\Sigma}\right)^n |y|,$$

is both reflexive and transitive. In order to prove Condition (iii) we need to proceed by induction on n: for the base case, $n = 0$, we get that $|x| \left(R^{\bigcirc}_{\Sigma}\right)^0 |y|$, which means that $|x| = |y|$ and therefore $x \sim_\Sigma y$. Hence, since $R^{\Box}_c$ is reflexive, $x \sim_\Sigma y$ and $\mathcal{M}_c, x \models \Box\varphi$ we get $M_c, y \models \varphi$.
For the inductive step let us assume that

$$\forall |z| \in W_\Sigma,\ |x| R^{\Box}_{\Sigma} |z| \text{ iff } \exists\, k \in \mathbb{N} \text{ s.t. } |x| \left(R^{\bigcirc}_{\Sigma}\right)^k |z|$$

and we try prove

$$|x| R^{\Box}_{\Sigma} |y| \text{ iff } \exists\, k+1 \in \mathbb{N} \text{ s.t. } |x| \left(R^{\bigcirc}_{\Sigma}\right)^{k+1} |y|.$$

Note that $|x| \left(R^{\bigcirc}_{\Sigma}\right)^{k+1} |y|$ can be rewritten as $|x| \left(R^{\bigcirc}_{\Sigma}\right)^k |z|$ and $|z| R^{\bigcirc}_{\Sigma} |y|$, for some $|z| \in W_\Sigma$. By induction hypothesis, this means that $|x| R^{\Box}_{\Sigma} |z|$ and $|z| R^{\bigcirc}_{\Sigma} |y|$. Now, since $R^{\Box}_{\Sigma}$ is transitive and the hypothesis $\mathcal{M}_c, x \models \Box\varphi$, we get that $\mathcal{M}_c, z \models \Box\varphi$ and, together with Axiom (28) from Table 4.1, we conclude that $\mathcal{M}_c, z \models \bigcirc\varphi$. Finally, since $R^{\bigcirc}_{\Sigma}$ satisfies Condition (vii) of filtration we get $\mathcal{M}_c, y \models \varphi$.

- **Condition (iv)**: Let $\Diamond\varphi \in \Sigma$ be a formula and $x \in W_c$ satisfying $\mathcal{M}_c, x \not\models \Diamond\varphi$. As in the previous case we proceed by induction on n. For the base case, $n = 0$ we have that $|x| \left(R^{\bigcirc}_{\Sigma}\right)^0 |y|$ iff $|x| = |y|$ and, therefore, $x \sim_\Sigma y$. Hence, since $R^{\Box}_c$ is reflexive, $x \sim_\Sigma y$ and $\mathcal{M}_c, x \not\models \Diamond\varphi$, we get $M_c, y \not\models \varphi$.
For the inductive step let us assume that

$$\forall |z| \in W_\Sigma,\ |x| R^{\Box}_{\Sigma} |z| \text{ iff } \exists\, k \in \mathbb{N} \text{ s.t. } |x| \left(R^{\bigcirc}_{\Sigma}\right)^k |z|$$

and we try prove

$$|x| R^{\Box}_{\Sigma} |y| \text{ iff } \exists\, k+1 \in \mathbb{N} \text{ s.t. } |x| \left(R^{\bigcirc}_{\Sigma}\right)^{k+1} |y|.$$

As in the previous case, $|x| \left(R^{\bigcirc}_{\Sigma}\right)^{k+1} |y|$ can be rewritten as $|x| \left(R^{\bigcirc}_{\Sigma}\right)^k |z|$ and $|z| R^{\bigcirc}_{\Sigma} |y|$, for some $|z| \in W_\Sigma$. By induction hypothesis, this means that $|x| R^{\Box}_{\Sigma} |z|$ and $|z| R^{\bigcirc}_{\Sigma} |y|$. Now, since $R^{\Box}_{\Sigma}$ is transitive and $\mathcal{M}_c, x \not\models \Diamond\varphi$,

we get that $\mathcal{M}_c, z \not\models \Diamond\varphi$. This, together with Axiom (28) from Table 4.1, allows us conclude that $\mathcal{M}_c, z \not\models \hat{\bigcirc}\varphi$. Finally, since $R_\Sigma^{\bigcirc}$ satisfies Condition (vii) of filtration we get $\mathcal{M}_c, y \not\models \varphi$.

$\boxtimes$

After showing that $\mathcal{M}_\Sigma$ is a filtration of $\mathcal{M}_c$ we define and prove the so called *filtration lemma*, which connects the satisfiability of a formula, $\varphi \in \Sigma$, in both the canonical and filtration models.

Lemma 4.15 (Filtration lemma)**.** *Let $\mathcal{M}_c = \langle W_c, R_c^{\Box}, R_c^{\bigcirc}, H_c, T_c\rangle$ be the canonical model and $\mathcal{M}_\Sigma = \langle W_\Sigma, R_\Sigma^{\Box}, R_\Sigma^{\bigcirc}, H_\Sigma, T_\Sigma\rangle$ its corresponding filtration model with respect to a theory, Σ, closed under subformulas. It holds that for every point $x \in \mathcal{M}_c$ and every formula $\varphi \in \Sigma$*

$$\mathcal{M}_c, x \models \varphi \textit{ iff } \mathcal{M}_\Sigma, |x| \models \varphi \textit{ and } \check{\mathcal{M}}_c, x \models \varphi \textit{ iff } \check{\mathcal{M}}_\Sigma, |x| \models \varphi.$$

Proof. We proceed by structural induction. Since the base case corresponds to $\varphi = p$, with p an atom. $\mathcal{M}_c, x \models p$ iff $x \in H_c(p)$ and, by the equivalence relation that induces the filtration, $|x| \in H_\Sigma(p)$ iff $\mathcal{M}_\Sigma, |x| \models p$. The case $\check{\mathcal{M}}_c, x \models p$ iff $\check{\mathcal{M}}_\Sigma, |x| \models p$ is also straightforward.

Now we apply structural induction on the subformulas. We will prove only the first part of the conjunction because the second part can be proved by following the same reasoning.

$$\begin{array}{lcll}
\mathcal{M}_c, x \models \alpha \wedge \beta & \Leftrightarrow & \mathcal{M}_c, x \models \alpha \text{ and } \mathcal{M}_c, x \models \beta & \text{(Def. 4.8)} \\
& \Leftrightarrow & \mathcal{M}_\Sigma, |x| \models \alpha \text{ and } \mathcal{M}_\Sigma, |x| \models \beta & \text{(Ind.)} \\
& \Leftrightarrow & \mathcal{M}_\Sigma, |x| \models \alpha \wedge \beta & \text{(Def. 4.8)}
\end{array}$$

$$\begin{array}{lcll}
\mathcal{M}_c, x \models \alpha \vee \beta & \Leftrightarrow & \mathcal{M}_c, x \models \alpha \text{ or } \mathcal{M}_c, x \models \beta & \text{(Def. 4.8)} \\
& \Leftrightarrow & \mathcal{M}_\Sigma, |x| \models \alpha \text{ or } \mathcal{M}_\Sigma, |x| \models \beta & \text{(Ind.)} \\
& \Leftrightarrow & \mathcal{M}_\Sigma, |x| \models \alpha \vee \beta & \text{(Def. 4.8)}
\end{array}$$

$$\begin{array}{lcll}
\mathcal{M}_c, x \models \alpha \rightarrow \beta & \Leftrightarrow & \begin{cases} (\mathcal{M}_c, x \not\models \alpha \text{ or } \mathcal{M}_c, x \models \beta) \\ \text{and} \\ (\check{\mathcal{M}}_c, x \not\models \alpha \text{ or } \check{\mathcal{M}}_c, x \models \beta) \end{cases} & \text{(Def. 4.8)} \\
& \Leftrightarrow & \begin{cases} (\mathcal{M}_\Sigma, |x| \not\models \alpha \text{ or } \mathcal{M}_\Sigma, |x| \models \beta) \\ \textit{and} \\ (\check{\mathcal{M}}_\Sigma, |x| \not\models \alpha \text{ or } \check{\mathcal{M}}_\Sigma, |x| \models \beta) \end{cases} & \text{(Ind.)} \\
& \Leftrightarrow & \mathcal{M}_\Sigma, |x| \models \alpha \rightarrow \beta & \text{(Def. 4.8)}
\end{array}$$

Now we proceed with the proof for modalities:

- $\varphi = \Box\alpha$ From left to right assume that $\mathcal{M}_c, x \models \Box\alpha$ but $\mathcal{M}_\Sigma, |x| \not\models \Box\alpha$. From $\mathcal{M}_\Sigma, |x| \not\models \Box\alpha$ we derive

$$\exists |y| \in W_\Sigma \text{ s.t. } |x| R_\Sigma^{\Box} |y| \text{ and } \mathcal{M}_\Sigma, |y| \not\models \alpha.$$

Hence, by Induction Hypothesis we obtain $\mathcal{M}_c, y \not\models \alpha$. On the other hand, since $\mathcal{M}_c, x \models \Box\alpha$ and $|x| R^{\Box}_{\Sigma} |y|$ then we conclude, by applying the condition (iii) of filtration, the contradiction $\mathcal{M}_c, y \models \alpha$.

From right to left, assume that $\mathcal{M}_\Sigma, |x| \models \Box\alpha$ but $\mathcal{M}_c, x \not\models \Box\alpha$. Since

$$\mathcal{M}_c, x \not\models \Box\alpha \Leftrightarrow \exists y \in W_c \text{ s.t. } x R^{\Box}_c y \text{ and } and \mathcal{M}_c, y \not\models \alpha.$$

thus, by applying the induction hypothesis on y we get that $\mathcal{M}_\Sigma, |y| \not\models \alpha$. On the other hand, from $x R^{\Box}_c y$ and condition (ii) we derive that

$$\exists z \in W_c | z | R^{\Box}_{\Sigma} |y| \text{ and either } |z| = |x| \text{ or } |z| = |\check{x}|.$$

If $|z| = |x|$, from $|x| R^{\Box}_{\Sigma} |y|$ and $\mathcal{M}_\Sigma, |x| \models \Box\alpha$ we reach the contradiction $\mathcal{M}_\Sigma, |y| \models \alpha$. Otherwise, if $|z| = |\check{x}|$ then the contradiction $\mathcal{M}_c, |y| \models \alpha$ follows from $|\check{x}| R^{\Box}_{\Sigma} |y|$ and $\mathcal{M}_\Sigma, |\check{x}| \models \Box\varphi$ (result derived by applying the property of persistence on x and then the induction hypothesis).

- $\varphi = \Diamond\alpha$

  From left to right, let us assume that $\mathcal{M}_c, x \models \Diamond\alpha$ but $\mathcal{M}_\Sigma, |x| \not\models \Diamond\alpha$. From the former assumption we get

  $$\exists y \in W_c \text{ s.t. } x R^{\Box}_c y \text{ and } \mathcal{M}_c, y \models \alpha$$

  and by Induction Hypothesis it holds that $\mathcal{M}_\Sigma, |y| \models \alpha$.

  Since x and y are related by means of $R^{\Box}_c$, the application of Condition (i) of filtration leads to the following result

  $$\exists z \in W_c \text{ s.t. } |x| R^{\Box}_{\Sigma} |z| \text{ and either } |z| = |y| \text{ or } |z| = |\check{y}|.$$

  Let us consider both cases:

  1. $|z| = |y|$: from $|x| R^{\Box}_{\Sigma} |y|$ and $\mathcal{M}_\Sigma, |y| \models \alpha$ we obtain that $\mathcal{M}_\Sigma, |x| \models \Diamond\alpha$.
  2. $|z| = |\check{y}|$: from $|x| R^{\Box}_{\Sigma} |\check{y}|$ and $\mathcal{M}_\Sigma, |\check{y}| \models \alpha$ (obtained by means of the property of persistence on y and the induction hypothesis) we conclude that $\mathcal{M}_\Sigma, |x| \models \Diamond\alpha$.

  Hence, in any case it holds that $\mathcal{M}_\Sigma, |x| \models \Diamond\alpha$, which contradicts our initial assumption.

  From right to left, assume that $\mathcal{M}_\Sigma, |x| \models \Diamond\alpha$ but $\mathcal{M}_c, x \not\models \Diamond\alpha$. From $\mathcal{M}_\Sigma, |x| \models \Diamond\alpha$ we obtain that

  $$\exists |y| \in W_\Sigma \text{ s.t. } |x| R^{\Box}_{\Sigma} |y| \text{ and } \mathcal{M}_c, y \models \alpha,$$

  which by Induction Hypothesis leads to derive $\mathcal{M}_c, y \models \alpha$. However, it contradicts the fact $\mathcal{M}_c, y \not\models \alpha$, which is derived from $|x| R^{\Box}_{\Sigma} |y|$, $\mathcal{M}_c, x \not\models \Diamond\alpha$ and Condition (iv) of filtration.

The proof for $\varphi = \bigcirc\alpha$ (resp. $\varphi = \hat{\bigcirc}\alpha$) is the same as for $\varphi = \Box\alpha$ (resp. $\varphi = \Diamond\alpha$) but we must use conditions (v)-(viii) instead of (i)-(iv).

⊠

4.2.5 Properties of filtration

We remind that $R_\Sigma^\Box$ is both reflexive, transitive by definition but we do not know anything about $R_\Sigma^\bigcirc$. We first check that seriality is preserved after filtration.

Lemma 4.16 (Seriality of $R_\Sigma^\bigcirc$)**.** *$R_\Sigma^\bigcirc$ is serial.*

Proof. Assume, for the sake of contradiction, that $R_\Sigma^\bigcirc$ is not serial, thus

$$\exists |x| \in W_\Sigma, \forall |y| \in W_\Sigma \ |x| \not R_\Sigma^\bigcirc |y|. \tag{4.12}$$

So, take $x' \in |x|$. Since $R_c^\bigcirc$ is serial then it holds that

$$\exists y \in W_c \text{ s.t. } x R_c^\bigcirc y$$

and it follows, from Condition (ii) of filtration, that

$$\exists z \in W_c \text{ s.t. } |x| R_\Sigma^\bigcirc |z| \text{ and either } |z| = |y| \text{ or } |z| = |\check{y}|$$

so we get a contradiction. ⊠

Unfortunately, $R_c^\bigcirc$ is functional (see Lemma 4.10), $|x| R_\Sigma^\bigcirc |y|$ might be not. To deal with this, we introduce the following lemma:

Lemma 4.17 ($R_\Sigma^\bigcirc$ is functional)**.** *Let $\bigcirc\varphi \in \Sigma$ be a temporal formula and $x \in W_c$ a system. The following conditions are equivalent:*

i) $\mathcal{M}_c, x \models \bigcirc\varphi$.

ii) $\forall y \in W_c \ \Big(|x| R_\Sigma^\bigcirc |y|$ *implies* $\mathcal{M}_c, y \models \varphi \Big)$

iii) $\exists y \Big(|x| R_\Sigma^\bigcirc |y|$ *and* $\mathcal{M}_c, y \models \varphi \Big)$

Conversely, the following three conditions are also equivalent:

iv) $\mathcal{M}_c, x \not\models \hat{\bigcirc}\varphi$.

v) $\exists y \Big(|x| R_\Sigma^\bigcirc |y|$ *and* $\mathcal{M}_c, y \not\models \varphi \Big)$

vi) $\forall y \in W_c \ \Big(|x| R_\Sigma^\bigcirc |y|$ *implies* $\mathcal{M}_c, y \not\models \varphi \Big)$

Proof.

- i)⇒ ii): assume that i) holds but ii) does not. Therefore

$$\exists y \in W_c \text{ s.t. } \left(|x| R^{\bigcirc}_{\Sigma} |y| \text{ and } \mathcal{M}_c, y \not\models \varphi\right).$$

  From i), $|x| R^{\bigcirc}_{\Sigma} |y|$ and the condition (iii) of filtration, we obtain the contradiction $\mathcal{M}_c, y \models \varphi$.

- ii)⇒ iii): Take $|x| \in W_c$, by Lemma 4.16 it follows that

$$\exists\ |y| \in W_{\Sigma} \text{ s. t. } |x| R^{\bigcirc}_{\Sigma} |y|.$$

  Thus, from ii) and $|x| R^{\bigcirc}_{\Sigma} |y|$ we get $\mathcal{M}_c, y \models \varphi$, that is

$$\exists y \in W_c \text{ s.t. } |x| R^{\bigcirc}_{\Sigma} |y| \text{ and } \mathcal{M}_c, y \models \varphi,$$

  which corresponds to iii).

- iii) ⇒ i): Suppose iii) but $\mathcal{M}_c\ x \not\models \bigcirc\varphi$. By applying the definition of $|x|\ R^{\bigcirc}_{\Sigma} |y|$ we obtain

$$\exists x' \in |x|\ \exists y' \in |y| \text{ s.t. } x' R^{\bigcirc}_{c} y'.$$

  It is easy to see that $\mathcal{M}_c, x' \not\models \bigcirc\varphi$ (because $x' \sim_{\Sigma} x$) and $\mathcal{M}_c, y' \models \varphi$ (because $y' \sim_{\Sigma} y$). Hence, $\mathcal{M}_c, x' \models \hat{\bigcirc}\varphi$ because $x' R^{\bigcirc}_{c} y'$. However the fact that $R^{\bigcirc}_{c}$ is functional, together with $\mathcal{M}_c, x' \models \hat{\bigcirc}\varphi$ and $\mathcal{M}_c, x' \not\models \bigcirc\varphi$, leads to a contradiction.

- iv) ⇒ vi): Suppose that iv) holds but (vi)) does not, that is,

$$\exists y \left(|x| R^{\bigcirc}_{\Sigma} |y| \text{ and } \mathcal{M}_c, y\ \models \varphi\right).$$

  From $|x|\ R^{\bigcirc}_{c}\ |y|$ and the condition (iv) of filtration, it follows the contradiction $\mathcal{M}_c, y \not\models \varphi$.

- vi) ⇒ v): Take $|x| \in W_{\Sigma}$, by Lemma 4.16 se obtain

$$\exists |y| \in W_{\Sigma}\ |x| R^{\bigcirc}_{\Sigma} |y|.$$

  From this conclusion and vi) we get $\mathcal{M}_c, y \not\models \varphi$, that is

$$\exists\ y \in W_c \text{ s.t. } |x| R^{\bigcirc}_{\Sigma} |y| \text{ and } \mathcal{M}_c, y \not\models \varphi,$$

  which is equivalent to v).

- v) $\Rightarrow$ iv): Suppose v) but $\mathcal{M}_c\ x \models \hat{\bigcirc}\varphi$. By applying the definition of $|x|\ R^{\bigcirc}_{\Sigma}|y|$ we obtain

$$\exists x' \in |x|\ \exists y' \in |y| \text{ s.t. } x' R^{\bigcirc}_{c} y'.$$

We can easily derive that $\mathcal{M}_c, x' \models \hat{\bigcirc}\varphi$ (because $x' \sim_\Sigma x$) and $\mathcal{M}_c, y' \not\models \varphi$ (because $y' \sim_\Sigma y$). Hence, $\mathcal{M}_c, x' \not\models \bigcirc\varphi$ because $x' R^{\bigcirc}_{c} y'$. However the fact that $R^{\bigcirc}_{c}$ is functional, together with $\mathcal{M}_c, x' \models \hat{\bigcirc}\varphi$ and $\mathcal{M}_c, x' \not\models \bigcirc\varphi$, leads to a contradiction.

$\boxtimes$

4.3 Conclusions

In this chapter we set the basis for the axiomatisation of THT. We define the concept of canonical model as well as the method of filtration for modal HT logics. We have proved that the axiomatic system is sound but, unfortunately, our proof of completeness still requires to adapt Goldblatt's argument about the use of Dummett's schema

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

from LTL to the case of THT as well as his consideration of the clusters induced by the relation $R^{\Box}_{\Sigma}$ (see Chapter 9 of [53] for a more detailed explanation about completeness proof for LTL). If we succeed in adapting those arguments, the next step would consist in proving the corresponding Truth Lemma for THT and, finally, its completeness, as we did in the case of $\mathrm{K_{HT}}$.

Chapter 5

Computing Temporal Equilibrium Models

In this chapter we study the algorithms available for computing temporal equilibrium/stable models. It must be noted that, as shown in Definition 3.3 and remarked in Observation 3.1, a temporal stable model (or TS-model) has the form of an LTL-interpretation, that is, an infinite sequence of propositional interpretations and, furthermore, it is quite usual that a theory has an infinite set of TS-models. Thus, the usual behaviour of ASP solvers enumerating finite stable models is not applicable here. Instead, temporal equilibrium models can be represented by mean of a Büchi automaton [16] (see Section 2.3.2), whose accepting language coincides with the set of TS-models.

As a result[1], we also present a pair of tools, `STeLP` and `ABSTEM` , which are based on the previously studied algorithms. Those tools are of different nature. While `STeLP` was firstly conceived for dealing with propositional *splittable* temporal logic programs, (a syntactic subclass of TEL) and then extended to the case of temporal programs with variables, `ABSTEM` is an implementation of the automata construction method presented in [18] and it is oriented to deal with propositional arbitrary temporal theories.

5.1 Splittable Temporal Logic Programs

Before presenting the definition of splittable temporal logic program, let us remind that some temporal expressions (for instance, rule 3.8) may have no temporal stable models due to, informally speaking, an "infinite dependence on the future." Fortunately, most ASP programs dealing with transition systems represent rules so that past does not depend on the future. This is what we called *future projected dependence* and can be captured by the following subclass of TLPs.

[1]The main results of this chapter have been published in [1] and [20]

Definition 5.1 (Splittable TLP (STLP))**.** *A TLP* Π *for signature* At *is said to be* splittable *if* Π *consists of rules of any of the forms:*

$$B \wedge N \rightarrow H \tag{5.1}$$
$$B \wedge \bigcirc B' \wedge N \wedge \bigcirc N' \rightarrow \bigcirc H' \tag{5.2}$$
$$\Box(B \wedge \bigcirc B' \wedge N \wedge \bigcirc N' \rightarrow \bigcirc H') \tag{5.3}$$

where B *and* B' *are conjunctions of atoms,* N *and* N' *are conjunctions of negative literals like* $\neg p$ *with* $p \in At$*, and* H *and* H' *are disjunctions of atoms.* ⊠

The set of rules of form (5.1) in Π will be denoted $ini_0(\Pi)$ and correspond to initial rules for situation 0. The rules of form (5.2) in Π will be represented as $ini_1(\Pi)$ and are initial rules for the transition between situations 0 and 1. Finally, the set of rules of form (5.3) is written $dyn(\Pi)$ and contains dynamic rules. Both in (5.2) and (5.3), we understand that operator $\bigcirc$ is actually shifted until it only affects to atoms – this is always possible due to equivalences (3.3), (3.4). By abuse of notation, we will also use the formulas B, B', N, N', H and H' as sets that respectively contain the atoms that occur in each respective formula.

An STLP is said to be *positive* if for all rules (5.1)-(5.3), N and N' are empty (an empty conjunction is equivalent to $\top$). An STLP is said to be *normal* if it contains no disjunctions, i.e., for all rules (5.1)-(5.1), H and H' are atoms. As an example of STLP take the program $\Pi_{5.1}$, which is shown below:

Example 5.1. *The following theory,* $\Pi_{5.1}$*, is an STLP:*

$$\neg a \wedge \bigcirc b \rightarrow \bigcirc a \tag{5.4}$$
$$\Box(a \rightarrow b) \tag{5.5}$$
$$\Box(\neg b \rightarrow \bigcirc a) \tag{5.6}$$

where (5.4) is an initial rule and (5.5),(5.6) are dynamic rules.

Notice that a rule of the form $\Box(B \wedge N \rightarrow H)$ (i.e., without $\bigcirc$ operator) is not splittable but it can be transformed into the equivalent pair of rules $B \wedge N \rightarrow H$ and $\Box(\bigcirc B \wedge \bigcirc N \rightarrow \bigcirc H)$ which are both splittable. For instance, (5.5) becomes the pair of rules:

$$a \rightarrow b \tag{5.7}$$
$$\Box(\bigcirc a \rightarrow \bigcirc b) \tag{5.8}$$

As an example, it is easy to check that the program

$$\Pi_1 = \{(5.4), (5.6), (5.7), (5.8)\}$$

is splittable, and its sets $ini_0(\Pi_1)$, $ini_1(\Pi_1)$ and $dyn(\Pi_1)$ are the following

$$\begin{aligned} ini_0(\Pi_1) &= \{(5.7)\} \\ ini_1(\Pi_1) &= \{(5.4)\} \\ dyn(\Pi_1) &= \{(5.8), (5.6)\}. \end{aligned}$$

In particular, in (5.4) we have the non-empty sets $B' = \{b\}$, $N = \{a\}$ and $H' = \{a\}$, whereas for (5.6) the sets are $N = \{b\}$, $H' = \{a\}$. On the other hand, the rule

$$\Box(\neg \bigcirc p \rightarrow p) \quad \wedge \quad \Box(\bigcirc p \rightarrow p)$$

is an example of a non-splittable program.

5.1.1 Splitting a temporal logic program

The most interesting feature of splittable TLPs is that we can adapt so-called *splitting* technique (see Section 2.1.3) to the temporal case and, hence, obtain their temporal equilibrium models in an incremental way by considering operator $\Box$ in dynamic rules $\Box r$ as as an infinite sequence of expressions $\bigcirc^i r$, one for each $i \geq 0$. Using (3.3),(3.4) we can shift $\bigcirc^i$ inside all connectives in r so that $\bigcirc^i r$ is equivalent to an initial rule resulting from prefixing any atom in r with $\bigcirc^i$. To put an example, if $r =$ (5.6) then $\bigcirc^2 r$ would correspond to $(\neg \bigcirc^2 b \rightarrow \bigcirc^3 a)$. We formalise this idea below.

Definition 5.2 (i-expansion of a rule)**.** *Given $i \geq 0$, the* i-expansion *of a dynamic rule $\Box r$, written $(\Box r)^i$, is a set of rules defined as:*

$$(\Box r)^i \stackrel{\text{def}}{=} \begin{cases} \emptyset & \text{if } i = 0 \text{ and } r \text{ contains some '}\bigcirc\text{'} \\ \{\bigcirc^j r \mid 0 \leq j \leq i-1\} & \text{if } i > 0 \text{ and } r \text{ contains some '}\bigcirc\text{'} \\ \{\bigcirc^j r \mid 0 \leq j \leq i\} & \text{otherwise} \end{cases}$$

If r is an initial rule, its i-expansion is defined as:

$$r^i \stackrel{\text{def}}{=} \begin{cases} \emptyset & \text{if } i = 0 \text{ and } r \text{ contains some '}\bigcirc\text{'} \\ r & \text{otherwise} \end{cases}$$

⊠

From Definition 5.2 we can easily conclude that, when Π is a splittable TLP, its program expansions have the form $\Pi^0 = ini_0(\Pi)$ and $\Pi^i = ini_0(\Pi) \cup ini_1(\Pi) \cup dyn(\Pi)^i$ for $i > 0$.

Proposition 5.1. *Given a splittable TLP Π for signature At and any $i \geq 0$:*

(i) *At^i is a splitting set for Π^ω;*

(ii) *and $b_{At^i}(\Pi^\omega) = \Pi^i$.*

⊠

Given any rule like r like (5.2) of (5.3) and a set of atoms X, we define its *simplification* $simp(r, X)$ as:

$$simp(r, X) \stackrel{\text{def}}{=} \begin{cases} \bigcirc B' \wedge \bigcirc N' \rightarrow \bigcirc H' & \text{if } B \subseteq X \text{ and } N \cap X = \emptyset \\ \top & \text{otherwise} \end{cases}$$

Given some LTL interpretation $\mathbf{T}$, let us define now the sequence of programs:

$$\Pi[\mathbf{T}, i] \quad \stackrel{\text{def}}{=} \quad e_{At^i}\left(\Pi^\omega \setminus \Pi^i\,,\ \mathbf{T}^i\right)$$

that is, $\Pi[\mathbf{T}, i]$ is the "simplification" of Π^ω by replacing atoms in At^i by their truth value with respect to $\mathbf{T}^i$. Then, we have:

Proposition 5.2.

$$\begin{aligned}\Pi[\mathbf{T}, 0] &= (dyn(\Pi)^\omega \setminus dyn(\Pi)^1) \cup \{simp(r, T_0) \mid r \in ini_1(\Pi) \cup dyn(\Pi)\}\\ \Pi[\mathbf{T}, i] &= (dyn(\Pi)^\omega \setminus dyn(\Pi)^{i+1}) \cup \ \{\bigcirc^i simp(r, T_i) \mid r \in dyn(\Pi)\}\end{aligned}$$

for any $i \geq 1$. ⊠

As we can see, programs $\Pi[\mathbf{T}, i]$ maintain most part of $dyn(\Pi)^\omega$ and only differ in simplified rules. Let us call these sets of simplified rules:

$$\begin{aligned}slice(\Pi, \mathbf{T}, 0) &\stackrel{\text{def}}{=} \Pi^0 = ini_0(\Pi)\\ slice(\Pi, \mathbf{T}, 1) &\stackrel{\text{def}}{=} \{simp(r, T_0) \mid r \in ini_1(\Pi) \cup dyn(\Pi)\}\\ slice(\Pi, \mathbf{T}, i+1) &\stackrel{\text{def}}{=} \{\bigcirc^i simp(r, T_i) \mid r \in dyn(\Pi)\} \qquad \text{for } i \geq 1\end{aligned}$$

Theorem 5.1 (Splitting sequence theorem adapted from [82])**.** *Let* $\langle \mathbf{T}, \mathbf{T}\rangle$ *be a model of a splittable TLP* Π. $\langle \mathbf{T}, \mathbf{T}\rangle$ *is a temporal equilibrium model of* Π *iff*

(i) $\mathbf{T}^0 = T_0$ *is a stable model of* $slice(\Pi, \mathbf{T}, 0) = \Pi^0 = ini_0(\Pi)$ *and*

(ii) $(\mathbf{T}^1 \setminus At^0)$ *is a stable model of* $slice(\Pi, \mathbf{T}, 1)$ *and*

(iii) $(\mathbf{T}^i \setminus At^{i-1})$ *is a stable model of* $slice(\Pi, \mathbf{T}, i)$ *for* $i \geq 2$.

⊠

As an example, let us take again program $\Pi_1 =$ (5.4), (5.7), (5.8), (5.6). The program $\Pi_1^0 = ini_0(\Pi_1) =$ (5.7) has the stable model $\mathbf{T}^0 = \emptyset = T_0$. Then we take $slice(\Pi, \mathbf{T}, 1) = \{simp(r, T_0) \mid r \in ini_1(\Pi) \cup dyn(\Pi)\}$ that corresponds to $\{(\bigcirc b \rightarrow \bigcirc a),\ (\bigcirc a \rightarrow \bigcirc b),\ (\top \rightarrow \bigcirc a)\}$ whose stable model is $\{\bigcirc a, \bigcirc b\} = (\mathbf{T}^1 \setminus At^0)$ so that $T_1 = \{a, b\}$. In the next step, $slice(\Pi, \mathbf{T}, 2) = \{\bigcirc simp(r, T_1) \mid r \in dyn(\Pi)\} = \{(\bigcirc^2 a \rightarrow \bigcirc^2 b), (\top)\}$ whose stable model is $\emptyset = (\mathbf{T}^2 \setminus At^1)$ so that $T_2 = \emptyset$. Then, we would go on with $slice(\Pi, \mathbf{T}, 3) = \{\bigcirc^2 simp(r, T_2) \mid r \in dyn(\Pi)\} = \{(\bigcirc^3 a \rightarrow \bigcirc^3 b), (\top \rightarrow \bigcirc^3 a)\}$ leading to $\{\bigcirc^3 a, \bigcirc^3 b\}$ that is $T_3 = \{a, b\}$ and so on.

5.1.2 Loop formulas for splittable temporal logic programs

Theorem 5.1 allows us building the temporal equilibrium models by considering an infinite sequence of finite ASP programs $slice(\Pi, \mathbf{T}, i)$. If we consider each program $\Pi' = slice(\Pi, \mathbf{T}, i+1)$ for signature $At^{i+1} \setminus At^i$ then, since it is a standard disjunctive ASP program, we can use the main result in [37] to compute its stable models by obtaining the classical models of a theory $\Pi' \cup LF(\Pi')$ where LF stands for *loop formulas*.

Given an ASP program Π we define its *(positive) dependency graph* $G(\Pi)$ where its vertices are At (the atoms in Π) and its edges are $E \subseteq At \times At$ so that $(p,p) \in E$ for any atom[2] p, and $(p,q) \in E$ if there is an ASP rule in Π like (5.1) with $p \in H$ and $q \in B$. A nonempty set L of atoms is called a *loop* of a program Π if, for every pair p, q of atoms in L, there exists a path from p to q in $G(\Pi)$ such that all vertices in the path belong to L. In other words, L is a loop of Π iff the subgraph of $G(\Pi)$ induced by L is strongly connected. Notice that reflexivity of $G(\Pi)$ implies that for any atom p, the singleton $\{p\}$ is also a loop.

Definition 5.3 (external support)**.** *Given an ASP program* Π *for signature* At*, the* external support *formula of a set of atoms* $Y \subseteq At$ *with respect to* Π*, written* $ES_\Pi(Y)$ *is defined by:*

$$\bigvee_{r \in R(Y)} \left(B \wedge N \wedge \bigwedge_{p \in H \setminus Y} \neg p \right)$$

where $R(Y) = \{r \in \Pi$ *like* (5.1) $\mid H \cap Y \neq \emptyset$ *and* $B \cap Y = \emptyset\}$. ⊠

Theorem 5.2 (from [37])**.** *Given a program* Π *for signature* At*, and a (classical) model* $X \subseteq At$ *of* Π *then* X *is a stable model of* Π *iff for every loop* Y *of* Π*,* X *satisfies* $\bigvee_{p \in Y} p \rightarrow ES_\Pi(Y)$ ⊠

This result can be directly applied to each finite ASP program $slice(\Pi, \mathbf{T}, i)$. As we have slice programs for $i = 0, 1, \ldots$, this means we would obtain an infinite sequence of classical theories (each program plus its loop formulas). Fortunately, these theories are not arbitrary. For situations $0, 1$, we may obtain loops induced by dependencies that are due to initial rules, but for $i \geq 2$ loops follow a *repetitive pattern*, so they can be easily captured using $\Box$ and $\bigcirc$ operators. Thus, we just need to consider loops for situations $0, 1, 2$ bearing in mind that any loop at level $i = 2$ will occur repeatedly from then on. Given a splittable TLP Π its associated dependency graph $G(\Pi)$ is generated from the expanded (ASP) program Π^2, so that its nodes are atoms in the signature At^2 and its loops are obtained from this finite program. For instance, given our example program Π_1, its graph $G(\Pi_1)$, shown in Figure 5.1, is obtained from the expanded program Π_1^2 and, as we can see, contains the loops $\{\bigcirc a, \bigcirc b\}$ plus $\{A\}$ for any $A \in At^2$.

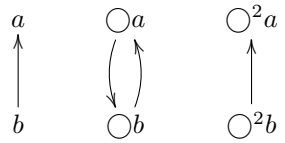


Figure 5.1: Graph $G(\Pi_1)$ (reflexive arcs are not displayed) corresponding to Π_1^2.

[2]The original formulation in [37] did not consider reflexive edges, dealing instead with the idea of paths of length 0.

Theorem 5.3 (from [3]). *Let* Π *be a splittable TLP and* $\mathbf{T}$ *an LTL model of* Π. *Then* $\langle \mathbf{T}, \mathbf{T} \rangle$ *is a temporal equilibrium model of* Π *iff* $\mathbf{T}$ *is an LTL model of the union of formulas* $LF(Y)$ *defined as:*

$$\begin{array}{rcll} Y^\vee & \rightarrow & ES_{ini_0(\Pi)}(Y) & \text{for any loop } Y \subseteq At^0 = At \\ Y^\vee & \rightarrow & ES_{ini_1(\Pi) \cup dyn(\Pi)^1}(Y) & \text{for any loop } Y \subseteq (At^1 \setminus At^0) \\ \Box\Big(Y^\vee & \rightarrow & ES_{dyn(\Pi)^2 \setminus dyn(\Pi)^1}(Y)\Big) & \text{for any loop } Y \subseteq (At^2 \setminus At^1) \end{array}$$

⊠

5.2 Temporal Quantified Equilibrium Logic

Syntactically, we consider function-free first-order languages $\mathcal{L} = \langle C, P \rangle$ built over a set of *constant* symbols, C, and a set of *predicate* symbols, P. Using $\mathcal{L}$, connectors and variables, an $\mathcal{L} = \langle C, P \rangle$*-formula* F is defined following the grammar:

$$F ::= p \mid \bot \mid F_1 \wedge F_2 \mid F_1 \vee F_2 \mid F_1 \rightarrow F_2 \mid$$
$$\bigcirc F \mid \ \Box F \mid \Diamond F \mid \forall x F(x) \mid \exists x F(x)$$

where $p \in P$ is an atom and x is a variable. The derived operators $\neg$, $\top$ and $\leftrightarrow$ are defined in the same way as in the propositional case as well as a theory is also defined as a set of formulas. An *atom* is any $p(t_1, \ldots, t_n)$ where $p \in P$ is a predicate with n-arity and each t_i is a term in its turn. We say that a term or a formula is *ground* if it does not contain variables. A $\mathcal{L}$*-sentence* or closed-formula is a formula without free-variables. A *temporal fact* is a construction of the form $\bigcirc^i A$ where A is an atom.

If D is a non-empty set, we denote by $At(D, P)$ the set of ground atomic sentences of the language $\langle D, P \rangle$. For the semantics, we will also define a mapping

$$\sigma \colon C \cup D \rightarrow D$$

such that $\sigma(d) = d$ for all $d \in D$.

A first-order LTL-interpretation is a structure $\langle (D, \sigma), \mathbf{T} \rangle$ where D is a non-empty set (the *domain*), σ is a mapping as defined above (the interpretation of constants) and $\mathbf{T}$ is an infinite sequence of sets of ground atoms $\mathbf{T} = \{T_i\}_{i \geq 0}$. Intuitively, $T_i \subseteq At(D, P)$ contains those ground atoms that are true at situation i. Given two LTL-interpretations $\mathbf{H}$ and $\mathbf{T}$ we say that $\mathbf{H}$ is *smaller than* $\mathbf{T}$, written $\mathbf{H} \leq \mathbf{T}$, when $H_i \subseteq T_i$ for all $i \geq 0$. As usual, $\mathbf{H} < \mathbf{T}$ stands for: $\mathbf{H} \leq \mathbf{T}$ and $\mathbf{H} \neq \mathbf{T}$. We define the ground temporal facts associated to $\mathbf{T}$ as follows: $Facts(\mathbf{T}) \stackrel{\text{def}}{=} \{\bigcirc^i p \mid p \in T_i\}$. It is easy to see that $\mathbf{H} \leq \mathbf{T}$ iff $Facts(\mathbf{H}) \subseteq Facts(\mathbf{T})$.

Definition 5.4. *A* temporal-here-and-there $\mathcal{L}$*-structure with static domains, or a* **TQHT**-structure*, is a tuple* $\mathbf{M} = \langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle$ *where* $\langle (D, \sigma), \mathbf{H} \rangle$ *and* $\langle (D, \sigma), \mathbf{T} \rangle$ *are two LTL-interpretations satisfying* $\mathbf{H} \leq \mathbf{T}$. ⊠

A **TQHT**-structure of the form $\mathbf{M} = \langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is said to be *total*. If $\mathbf{M} = \langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle$ is a **TQHT**-structure and k any positive integer, we

denote by $(\mathbf{M}, k) = \langle (D, \sigma), (\mathbf{H}, k), (\mathbf{T}, k) \rangle$ the temporal-here-and-there $\mathcal{L}$-structure with $(\mathbf{H}, k) = \{H_i\}_{i \geq k}$ and $(\mathbf{T}, k) = \{T_i\}_{i \geq k}$. The satisfaction relation for $\mathbf{M} = \langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle$ is defined recursively forcing us to consider formulas from $\langle C \cup D, P \rangle$. Formally, if φ is an $\mathcal{L}$-sentence for the atoms in $At(C \cup D, P)$, then:

- If $\varphi = p(t_1, \ldots, t_n) \in At(C \cup D, P)$, then

$$\begin{array}{rcl} \mathbf{M} \models p(t_1, \ldots, t_n) & \text{iff} & p(\sigma(t_1), \ldots, \sigma(t_n)) \in H_0. \\ \mathbf{M} \models t = s & \text{iff} & \sigma(t) = \sigma(s) \end{array}$$

- $\mathbf{M} \not\models \bot$
- $\mathbf{M} \models \varphi \wedge \psi$ iff $\mathbf{M} \models \varphi$ and $\mathbf{M} \models \psi$.
- $\mathbf{M} \models \varphi \vee \psi$ iff $\mathbf{M} \models \varphi$ or $\mathbf{M} \models \psi$.
- $\mathbf{M} \models \varphi \rightarrow \psi$ iff $\langle (D, \sigma), w, \mathbf{T} \rangle \not\models \varphi$ or $\langle (D, \sigma), w, \mathbf{T} \rangle \models \psi$ for all $w \in \{\mathbf{H}, \mathbf{T}\}$
- $\mathbf{M} \models \bigcirc \varphi$ if $(\mathbf{M}, 1) \models \varphi$.
- $\mathbf{M} \models \Box \varphi$ if $\forall j \geq 0$, $(\mathbf{M}, j) \models \varphi$
- $\mathbf{M} \models \Diamond \varphi$ if $\exists j \geq 0$, $(\mathbf{M}, j) \models \varphi$
- $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \forall x \varphi(x)$ iff $\langle (D, \sigma), w, \mathbf{T} \rangle \models \varphi(d)$ for all $d \in D$ and for all $w \in \{\mathbf{H}, \mathbf{T}\}$.
- $\mathbf{M} \models \exists x \varphi(x)$ iff $\mathbf{M} \models \varphi(d)$ for some $d \in D$.

The resulting logic is called *Quantified Temporal Here-and-There Logic with static domains*, and denoted by **SQTHT** or simply by **QTHT**. It is not difficult to see that, if we restrict to total **TQHT**-structures, $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle \models \varphi$ iff $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle \models \varphi$ in first-order LTL. Furthermore, the following property can be easily checked by structural induction.

Proposition 5.3. *For any formula φ, if $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \varphi$, then $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle \models \varphi$* ⊠

A *theory* Γ is a set of $\mathcal{L}$-sentences. An interpretation $\mathbf{M}$ is a model of a theory Γ, written $\mathbf{M} \models \Gamma$, if it satisfies all the sentences in Γ.

Definition 5.5 (temporal equilibrium model)**.** *A* temporal equilibrium *model of a theory Γ is a total model $\mathbf{M} = \langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ of Γ such that there is no $\mathbf{H} < \mathbf{T}$ satisfying $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \Gamma$.* ⊠

If $\mathbf{M} = \langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of a theory Γ, we say that the First-Order LTL interpretation $\langle (D, \sigma), \mathbf{T} \rangle$ is a *temporal stable model* of Γ. We write $TSM(\Gamma)$ to denote the set of temporal stable models of Γ. The set of *credulous consequences* of a theory Γ, written $CredFacts(\Gamma)$ contains all the temporal facts that occur at some temporal stable model of Γ, that is:

$$CredFacts(\Gamma) \stackrel{\text{def}}{=} \bigcup_{\langle (D,\sigma), \mathbf{T} \rangle \in TSM(\Pi)} Facts(\mathbf{T})$$

A property of TEL directly inherited from Equilibrium Logic (see Proposition 5 in [99]) is the following:

Proposition 5.4 (cummulativity for negated formulas)**.** *Let Γ be some theory and let $\neg\varphi$ be some formula such that $\mathbf{M} \models \neg\varphi$ for all temporal equilibrium models of Γ. Then, the theories Γ and $\Gamma \cup \{\neg\varphi\}$ have the same set of temporal equilibrium models.* ⊠

It is well-known that stable models (and so Equilibrium Logic) do not satisfy cummulativity in the general case: that is, if a formula is satisfied in all the stable models, adding it to the program may vary the consequences we obtain. However, when we deal with negated formulas, Proposition 5.4 tells us that cummulativity is guaranteed.

Definition 5.6. *A* first-order splittable temporal logic program *is a finite set of sentences like*

$$\varphi = \forall x_1 \forall x_2 \ldots \forall x_n \psi,$$

where ψ is a splittable temporal formula with $x_1, x_2, \ldots, x_n$ free variables. ⊠

For an example including variables, let us consider the following example:

Example 5.2. *Suppose we have a set of cars placed at different cities and, at each transition, we can drive a car from one city to another in a single step, provided that there is a road connecting them.* ⊠

The encoding of this example can be represented as an STLP $\Pi_{5.2}$ whose logical representation corresponds to:

$$\Box(\ Driveto(x,a) \rightarrow \bigcirc At(x,a)\ ) \tag{5.9}$$

$$\Box(\ At(x,a) \wedge Road(a,b) \rightarrow Driveto(x,b) \vee NoDriveto(x,b)\ ) \tag{5.10}$$

$$\Box(\ At(x,a) \wedge \neg \bigcirc NoAt(x,a) \rightarrow \bigcirc At(x,a)\ ) \tag{5.11}$$

$$\Box(\ At(x,b) \wedge City(a) \wedge a \neq b \rightarrow NoAt(x,a)\ ) \tag{5.12}$$

$$\Box(\ At(x,a) \wedge At(x,b) \wedge a \neq b \rightarrow \bot\ ) \tag{5.13}$$

Remember that all rule variables are implicitly universally quantified. For simplicity, we assume that inequality is a predefined predicate.

Proposition 5.5. *Any normal first-order positive STLP Π has a unique temporal stable model $\langle (D, \sigma), \mathbf{T} \rangle$ which coincides with its $\leq$-least LTL-model. We denote $LM(\Pi) = Facts(\mathbf{T})$.* ⊠

5.2.1 Safe Variables and Domain Independence

In this section we prove the the typical safety condition imposed in ASP first-order representation (see Definition 2.12) can be applied to the temporal case.

Definition 5.7 (Safety condition)**.** *A splittable temporal formula φ of type (5.1), (5.2) or (5.3) is said to be* safe *if, for any variable x occurring in φ, there exists an atomic formula p in B or B' such that x occurs in p. A formula $\forall x_1 \forall x_2 \ldots \forall x_n \psi$ is safe if the splittable temporal formula ψ is safe.* ⊠

For instance, rules (5.9)-(5.13) are safe. A simple example of unsafe rule is the splittable temporal formula:

$$\top \rightarrow P(x) \tag{5.14}$$

where x does not occur in the positive body (in fact, the rule body is empty). Although an unsafe rule not always leads to lack of domain independence (see examples in Section 2.1.7) it is frequently the case. For instance, if a program contains (5.14) and we add a new fresh constant c to the signature, stable models will contain the ground fact $P(c)$, something that was obviously impossible before the addition of constant c not mentioned before. We prove next that domain independence is, in fact, guaranteed for safe STLP's.

A variable assignment μ in a universe (D, σ) is a mapping from the set of variables to D. If $\varphi \in \mathcal{L}$ has free-variables, φ^μ is the closed formula obtained by replacing every free variable x by $\mu(x)$. From now on, if $\mathbf{T} = \{T_i\}_{i\geq 0}$ with $T_i \subseteq At(D, P)$, we denote by $\mathbf{T}|_C$ the sequence of sets defined by $\mathbf{T}|_C = \{T_i|C\}_{i\geq 0}$, where each $T_i|_C = T_i \cap At(\sigma(C), P)$ is the subset of T_i whose atoms contain terms only from $\sigma(C)$, that is, those atoms exclusively formed with elements in the universe that are images of syntactic constants in C.

Lemma 5.1. *Let φ be a splittable temporal formula and μ a variable assignment in (D, σ). If φ is safe, then if follows that:*

$$\langle (D, \sigma), \mathbf{T}, \mathbf{T}\rangle \models \varphi^\mu \textit{ iff } \langle (D, \sigma), \mathbf{T}|_C, \mathbf{T}\rangle \models \varphi^\mu.$$

Proof. First of all, take $\varphi = B \wedge N \rightarrow H$ of type 5.1 and suppose that $\langle \mathbf{T}, \mathbf{T}\rangle \models \varphi^\mu$ but $\langle \mathbf{T}|_C, \mathbf{T}\rangle \not\models \varphi^\mu$. This means that $\langle \mathbf{T}|_C, \mathbf{T}\rangle \models B^\mu \wedge N^\mu$ and $\langle \mathbf{T}|_C, \mathbf{T}\rangle \not\models H^\mu$. Since $\langle \mathbf{T}, \mathbf{T}\rangle \models H^\mu$, there exists an atomic formula q in H such that $\langle \mathbf{T}, \mathbf{T}\rangle \models q^\mu$ but $\langle \mathbf{T}|_C, \mathbf{T}\rangle \not\models q^\mu$. So we have a variable x in q with $\mu(x) \notin \sigma(C)$. As φ is safe, we know that x occurs in an atomic formula p in B. Then $\langle \mathbf{T}|_C, \mathbf{T}\rangle \not\models p^\mu$ and $\langle \mathbf{T}|_C, \mathbf{T}\rangle \not\models B^\mu$ which yields a contradiction.

If φ is of type (5.2), we use a similar argument.

Finally, take $\varphi = \Box(B \wedge \bigcirc B' \wedge N \wedge \bigcirc N' \rightarrow \bigcirc H') = \Box\psi$ of type (5.3) and suppose that $\langle \mathbf{T}, \mathbf{T}\rangle \models \varphi^\mu$ but $\langle \mathbf{T}|_C, \mathbf{T}\rangle \not\models \varphi^\mu$. There exists $i \geq 0$ such that $\langle T_i, T_i\rangle \models \psi^\mu$ and $\langle T_i \cap \sigma(C), T_i\rangle \not\models \psi^\mu$. We then have that $\langle T_i \cap \sigma(C), T_i\rangle \models B^\mu \wedge (\bigcirc B')^\mu \wedge N^\mu \wedge (\bigcirc N')^\mu$ and $\langle T_i \cap \sigma(C), T_i\rangle \not\models (\bigcirc H')^\mu$. Using the fact that φ is safe and the same argument as above, we find an atomic formula p in B or B' such that $\langle T_i \cap \sigma(C), T_i\rangle \not\models p^\mu$ which implies $\langle T_i \cap \sigma(C), T_i\rangle \not\models B^\mu \wedge (\bigcirc B')^\mu$ and leads to contradiction. The other implication follows directly from proposition 5.3. ⊠

Proposition 5.6. *For any safe sentence $\varphi = \forall x_1 \forall x_2 \ldots \forall x_n \psi$*

$$\langle (D, \sigma), \mathbf{T}, \mathbf{T}\rangle \models \varphi \quad \textit{iff} \quad \langle (D, \sigma), \mathbf{T}|_C, \mathbf{T}\rangle \models \varphi.$$

Proof. Proceed by induction over the length of the prefix. If $n = 0$, we can take any μ assignment of variables and apply Lemma 5.1 on $\varphi = \varphi^\mu$. So take $\varphi = \forall x_1 \ldots \forall x_n \psi$ of length n and suppose that the result is true for any universal safe sentence whose prefix has length at most $n - 1$. If $\langle (D, \sigma), \mathbf{T}, \mathbf{T}\rangle \models \varphi$, put $\varphi = \forall x_1 \alpha(x_1)$ with $\alpha(x_1) = \forall x_2 \ldots \forall x_n \psi$. For any $d \in D$, we know that

$\langle (D,\sigma), \mathbf{T}, \mathbf{T}\rangle \models \alpha(d)$ and we have to show that $\langle (D,\sigma), \mathbf{T}|_C, \mathbf{T}\rangle \models \alpha(d)$. The induction hypothesis and the fact that $\alpha(d)$ is a safe sentence whose prefix has length smaller or equal than $n-1$ finishes the proof. ⊠

Theorem 5.4. *If φ is a safe sentence and $\langle (D,\sigma), \mathbf{T}, \mathbf{T}\rangle$ is a temporal equilibrium model of φ, then $\mathbf{T}|_C = \mathbf{T}$ and $T_i \subseteq At(\sigma(C), P)$ for any $i \geq 0$.*

Proof. If φ is a safe sentence and $\langle (D,\sigma), \mathbf{T}, \mathbf{T}\rangle$ is a temporal equilibrium model of φ, we have that $\langle (D,\sigma), \mathbf{T}|_C, \mathbf{T}\rangle \models \varphi$ by proposition 5.6. The definition of temporal equilibrium model implies that $\mathbf{T}|_C = \mathbf{T}$ and $T_i \subseteq At(\sigma(C), P)$ for any $i \geq 0$. ⊠

Let (D,σ) be a domain and $D' \subseteq D$ a finite subset; the grounding over D' of a sentence φ, denoted by $\mathrm{Gr}_{D'}(\varphi)$, is defined recursively

$$
\begin{aligned}
\mathrm{Gr}_{D'}(p) &\stackrel{\text{def}}{=} p\text{, where } p \text{ denotes any atomic formula}\\
\mathrm{Gr}_{D'}(\varphi_1 \odot \varphi_2) &\stackrel{\text{def}}{=} \mathrm{Gr}_{D'}(\varphi_1) \odot \mathrm{Gr}_{D'}(\varphi_2)\text{, with } \odot \text{ any binary operator in } \{\wedge, \vee, \rightarrow\}\\
\mathrm{Gr}_{D'}(\forall x \varphi(x)) &\stackrel{\text{def}}{=} \bigwedge_{d \in D'} \mathrm{Gr}_{D'}\varphi(d)\\
\mathrm{Gr}_{D'}(\exists x \varphi(x)) &\stackrel{\text{def}}{=} \bigvee_{d \in D'} \mathrm{Gr}_{D'}\varphi(d)\\
\mathrm{Gr}_{D'}(\bigcirc \varphi) &\stackrel{\text{def}}{=} \bigcirc \mathrm{Gr}_{D'}(\varphi)\\
\mathrm{Gr}_{D'}(\Box \varphi) &\stackrel{\text{def}}{=} \Box \mathrm{Gr}_{D'}(\varphi)\\
\mathrm{Gr}_{D'}(\Diamond \varphi) &\stackrel{\text{def}}{=} \Diamond \mathrm{Gr}_{D'}(\varphi)
\end{aligned}
$$

Proposition 5.7. *Given any non empty finite set D and any sentence φ:*

$$\langle (D,\sigma), \mathbf{H}, \mathbf{T}\rangle \models \varphi \textit{ iff } \langle (D,\sigma), \mathbf{H}, \mathbf{T}\rangle \models \mathrm{Gr}_D(\varphi).$$

⊠

Lemma 5.2. *Let $\varphi(x)$ be a safe splittable temporal formula of type (5.1), (5.2) or (5.3) and take $\langle (D,\sigma), \mathbf{H}, \mathbf{T}\rangle$ be such that $\mathbf{T} = \mathbf{T}|C$. Then, for any $d \in D \setminus \sigma(C)$ we have:*

$$\langle (D,\sigma), \mathbf{H}, \mathbf{T}\rangle \models \varphi(d).$$

Proof. First of all, suppose that $\varphi(x)$ is of type (5.1):

$$B \wedge N \rightarrow H$$

and take $d \in D \setminus \sigma(C)$ and $w \in \{\mathbf{H}, \mathbf{T}\}$ such that $\langle (D,\sigma), w, \mathbf{T}\rangle \not\models \varphi(d)$. This implies that $\langle (D,\sigma), w, \mathbf{T}\rangle \models B(d) \wedge N(d)$ but $\langle (D,\sigma), w, \mathbf{T}\rangle \not\models H(d)$. $\varphi(x)$ is safe so there must be an atom p in B such that x has an occurrence in p. Since $T_0 \subseteq \mathrm{At}(\sigma(C), P)$, it is clear that $\langle (D,\sigma), w, \mathbf{T}\rangle \not\models p(d)$, so $\langle (D,\sigma), w, \mathbf{T}\rangle \not\models B(d)$ which yields a contradiction.

The proof for the case of $\varphi(x)$ being of type (2) and (3) is similar. ⊠

Lemma 5.3. *Let $\varphi(x) = \forall x_1 \forall x_2 \ldots \forall x_n \psi$ with ψ a splittable temporal formula and such that $\varphi(x)$ has no other free variables than x. Let $\mathbf{M} = \langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle$ be such that $\mathbf{T} = \mathbf{T}|C$. Then, if $\forall x \varphi(x)$ is safe, we have that:*

$$\mathbf{M} \models \forall x \varphi(x) \textit{ iff } \mathbf{M} \models \bigwedge_{c \in C} \varphi(c).$$

Proof. From left to right, just note that if $\mathbf{M} \models \forall x \varphi(x)$ but $\mathbf{M} \not\models \varphi(c)$, for some $c \in C$, we would have that $\mathbf{M} \not\models \varphi(\sigma(c))$ which would yield a contradiction.

For right to left, we can proceed by induction in n. If $n = 0$, then $\varphi(x)$ is in the case of the previous lemma for any $d \in D \backslash \sigma(C)$, so $\mathbf{M} \models \forall x \varphi(x)$ whenever $\mathbf{M} \models \bigwedge_{c \in C} \varphi(c)$. Now, suppose the result is true for any prenex formula with length up to $n - 1$ and take $\varphi(x) = \forall x_1 \forall x_2 \ldots \forall x_n \psi(x, x_1, \ldots, x_n)$ such that $\mathbf{M} \models \bigwedge_{c \in C} \varphi(c)$. In order to conclude the result, it only rests to show that $\mathbf{M} \models \varphi(d)$ for any $d \in D \setminus \sigma(C)$. Notice that $\varphi(d) = \forall x_1 \alpha(x_1)$ with $\alpha(x_1) = \forall x_2 \ldots \forall x_n \psi(d, x_1, x_2, \ldots, x_n)$. Since we can apply the induction hypothesis on $\alpha(x_1)$, it will be sufficient to prove that:

$$\mathbf{M} \models \bigwedge_{c \in C} \alpha(c).$$

Now fix any $c \in C$ and take into account that

$$\mathbf{M} \models \varphi(c') = \forall x_1 \forall x_2 \ldots \forall x_n \psi(c', x_1, x_2 \ldots, x_n)$$

for all $c' \in C$, so we can replace x_1 by any constant in C, including c, and it holds that:

$$\mathbf{M} \models \forall x_2 \ldots \forall x_n \psi(c', c, x_2, \ldots, x_n), \text{ for any } c' \in C$$

Observe that we can apply the induction hypothesis on $\beta(z)$, where

$$\beta(z) = \forall x_2 \ldots \forall x_n \psi(z, c, x_2, \ldots, x_n)$$

and then $\mathbf{M} \models \forall z \beta(z)$. In particular $\mathbf{M} \models \beta(d)$ which completes the proof since $\beta(d) = \alpha(c)$. ⊠

Theorem 5.5. *If $\varphi = \forall x_1 \forall x_2 \ldots \forall x_n \psi$ is a safe splittable temporal sentence and $\mathbf{M} = \langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle$ such that $\mathbf{T} = \mathbf{T}|C$, then*

$$\mathbf{M} \models \varphi \quad \textit{iff} \quad \mathbf{M} \models \mathrm{Gr}_C(\varphi).$$

Proof. From left to right, suppose that $\mathbf{M} \models \varphi$. By Proposition 5.7, we know that $\mathbf{M} \models \mathrm{Gr}_D(\varphi)$. The result follows since $\sigma(C) \subseteq D$ and $\mathrm{Gr}_C(\varphi) = \mathrm{Gr}_{\sigma(C)}(\varphi)$.

Now, from the right to left direction, take $\varphi = \forall x_1 \forall x_2 \ldots \forall x_n \psi$ a safe splitable temporal sentence and suppose that $\mathbf{M} \models \mathrm{Gr}_C(\varphi)$. Again, we can proceed by induction in n. If $n = 0$, then φ is quantifier free so $\mathrm{Gr}_C(\varphi) = \varphi$. Suppose the result is true for any safe splitable sentence with length up to $n - 1$ and put $\varphi = \forall x_1 \alpha(x_1)$ with $\alpha(x_1) = \forall x_2 \ldots \forall x_n \psi(x_1, x_2, \ldots, x_n)$. Notice that $\alpha(x_1)$ is a safe formula that has no more free variables than x_1, so, if we apply Lemma 5.3, it will be sufficient to show that $\mathbf{M} \models \bigwedge_{c \in C} \alpha(c)$. Since we are supposing that

$$\mathbf{M} \models \mathrm{Gr}_C(\varphi) = \bigwedge_{c \in C} \mathrm{Gr}_C(\alpha(c)),$$

and we can apply the induction hypothesis on any $\alpha(c)$ with $c \in C$, it follows that $\mathbf{M} \models \bigwedge_{c \in C} \alpha(c)$ and this completes the proof. ⊠

Theorem 5.6. *If φ is a safe splittable temporal sentence, then $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of φ iff $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of $\mathrm{Gr}_C(\varphi)$.*

Proof. Suppose that $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of φ and $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \mathrm{Gr}_C(\varphi)$. Since φ is safe, we know by Theorem 5.4 that $\mathbf{T} = \mathbf{T}|C$ so, applying Theorem 5.5, it follows that $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \varphi$ and $\mathbf{H} = \mathbf{T}$. This shows that $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is also a temporal equilibrium model of $\mathrm{Gr}_C(\varphi)$, The other implication follows directly from the fact that $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \varphi$ implies $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \mathrm{Gr}_C(\varphi)$. ⊠

Theorem 5.7 (Domain independence)**.** *Let φ be safe splittable temporal sentence. Suppose we expand the language $\mathcal{L}$ by considering a set of constants $C' \supseteq C$. A total* **QTHT***-model $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of $\mathrm{Gr}_{C'}(\varphi)$ if and only if it is a temporal equilibrium model of $\mathrm{Gr}_C(\varphi)$.*

Proof of Theorem 5.7. Let us show that the following assertions are equivalent:

1. $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of $\mathrm{Gr}_C(\varphi)$
2. $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of φ
3. $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of $\mathrm{Gr}_{C'}(\varphi)$

Taking into account the previous theorem, we only have to prove the equivalence of 2 and 3. Suppose that $\langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$ is a temporal equilibrium model of φ and $\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \mathrm{Gr}_{C'}(\varphi)$. Because of Theorem 5.4, we have that $\mathbf{T} = \mathbf{T}|C \subseteq \mathbf{T}|C'$ and an obvious extension of Theorem 5.5 to C', implies that

$$\langle (D, \sigma), \mathbf{H}, \mathbf{T} \rangle \models \varphi$$

and so $\mathbf{H} = \mathbf{T}$. This shows that 2 implies 3. The other implication ($3. \Longrightarrow 2.$) follows directly. ⊠

Theorem 5.7 guarantees that, since the program $\Pi_{5.2}$ is safe, $Gr_{(D)}(\Pi_{5.2})$ preserves the set of temporal equilibrium models for whatever domain we consider. Suppose we add the following set of facts:

```
Car(1), Car(2), City(Lisbon), City(Madrid), City(Paris),
City(Boston), City(Ny), City(Nj), Road(Lisbon,Madrid),
Road(Madrid,Lisbon), Road(Madrid,Paris),
Road(Paris,Madrid), Road(Boston,Ny), Road(Ny,Boston),
Road(Ny,Nj), Road(Nj,Ny), At(1,Madrid), At(2,Ny).
```

The domain consists of the set of constants

$$\{1,\ 2,\ Lisbon,\ Madrid,\ Paris,\ Boston,\ Ny,\ Nj\}$$

and, in this case, Gr_D((5.10)) generates 512 ground rules, some of them like:

$$\Box(\ At(1,Nj) \wedge Road(Nj,Ny) \rightarrow Driveto(1,Ny) \vee NoDriveto(1,Ny). \quad (5.15)$$

This rule intuitively means that whenever the car 1 be in New Jersey, since the ground atom `Road(Ny,Nj)` states a road between New Jersey y New York, the driver can choose between driving to N. York or not. However, we can conclude that, since `At(1,Madrid)` places the car in Madrid at instant 0 and N. Jersey can never be reached from Madrid, `At(1,Nj)` is never derived from the program. Hence the body of Rule (5.15) is false in every TEL model and, as a consequence, it can be removed from $Gr_{(D)}(\Pi_{5.2})$.

In next section we present a technique for grounding safe temporal programs that avoids generating rules like (5.15). This technique is based on the construction of a positive normal ASP program with variables. The least model of this program can be obtained by the ASP grounder[3] `DLV` and it can be used afterwards to provide the variable substitutions to be performed on the STLP.

5.2.2 Derivable ground facts

The method is based on the idea of *derivable* ground temporal facts for an STLP Π. This set, call it Δ, will be an upper estimation of the credulous consequences of the program, that is, $CredFacts(\Pi) \subseteq \Delta$. Of course, the ideal situation would be that $\Delta = CredFacts(\Pi)$, but the set $CredFacts(\Pi)$ requires the temporal stable models of Π and these (apart from being infinite sequences) will not be available at grounding time. In the worst case, we could choose Δ to contain the whole set of possible temporal facts, but this would not provide relevant information to improve grounding. So, we will try to obtain some superset of $CredFacts(\Pi)$ as small as possible, or if preferred, to obtain the largest set of *non-derivable* facts we can find. Note that a non-derivable fact $\bigcirc^i p \notin \Delta$ satisfies that $\bigcirc^i p \notin CredFacts(\Pi)$ and so, by Proposition 5.4, $\Pi \cup \{\neg \bigcirc^i p\}$ is equivalent to Π, that is, both theories have the same set of temporal equilibrium models. This information can be used to simplify the ground program either by removing rules or literals.

We begin defining several transformations on STLP's. For any temporal rule r, we define $r^\wedge$ as the set of rules:

- If r has the form (5.1) then

$$r^\wedge \stackrel{\text{def}}{=} \{B \rightarrow p \mid \text{atom } p \text{ occurs in } H\}$$

- If r has the form (5.2) then

$$r^\wedge \stackrel{\text{def}}{=} \{B \wedge \bigcirc B' \rightarrow \bigcirc p \mid \text{atom } p \text{ occurs in } H'\}$$

- If r has the form (5.3) then

$$r^\wedge \stackrel{\text{def}}{=} \{\Box(B \wedge \bigcirc B' \rightarrow \bigcirc p) \mid \text{atom } p \text{ occurs in } H'\}$$

[3] Or any other ASP grounder, such as `gringo`, respecting the safety condition of Definition 2.12.

In other words, $r^\wedge$ results from removing all negative literals in r and, informally speaking, transforming disjunctions in the head into conjunctions, so that $r^\wedge$ will imply *all* the original disjuncts in the disjunctive head of r. It is interesting to note that for any rule r with an empty head ($\bot$) this definition implies $r^\wedge = \emptyset$. Program $\Pi^\wedge$ is defined as the union of $r^\wedge$ for all rules $r \in \Pi$. As an example, $\Pi^\wedge_{5.1}$ consists of the rules:

$$\begin{aligned} \bigcirc b &\rightarrow \bigcirc a \\ a &\rightarrow b \\ \Box(\bigcirc a &\rightarrow \bigcirc b) \\ \Box(\top &\rightarrow \bigcirc a) \end{aligned}$$

whereas $\Pi^\wedge_{5.2}$ would be the program:

$$\Box(\ Driveto(x,a) \rightarrow \bigcirc At(x,a)\ ) \tag{5.16}$$
$$\Box(\ At(x,a) \wedge Road(a,b) \rightarrow Driveto(x,b)\ ) \tag{5.17}$$
$$\Box(\ At(x,a) \wedge Road(a,b) \rightarrow NoDriveto(x,b)\ ) \tag{5.18}$$
$$\Box(\ At(x,a) \rightarrow \bigcirc At(x,a)\ ) \tag{5.19}$$
$$\Box(\ At(x,b) \wedge City(a) \wedge a \neq b \rightarrow NoAt(x,a)\ ) \tag{5.20}$$

If we look carefully at this example program, we are now moving each car x so that it will be at several cities at the same time (constraint (5.13) has been removed) and, at each step, it will additionally locate car x in all adjacent cities to the previous ones "visited" by x. In this way, if we conclude $\bigcirc^i At(x,a)$ from this program this is actually representing that car x *can reach* city a in i steps or less. In some sense, $\Pi^\wedge$ looks like a heuristic simplification[4] of the original problem obtained by removing some constraints (this is something common in the area of Planning in Artificial Intelligence).

Notice that, by definition, $\Pi^\wedge$ is always a positive normal STLP and, by Proposition 5.5, it has a unique temporal stable model, $LM(\Pi^\wedge)$. We start proving a very basic result for non-temporal programs that will be used, as an auxiliary lemma, in the subsequent proofs.

Lemma 5.4. *If T is any equilibrium model of a (non temporal) program Π with rules of type* (5.1)*, then $T \subseteq J$, where J is any model of the normal positive program $\Pi^\wedge$.*

Proof. We will prove the result by showing that the interpretation $\langle T \cap J, T\rangle \models \Pi$ and, in consequence, $T \cap J = T$ by the minimality of T.

Let $B \wedge N \rightarrow H$ of the form (5.1) be an arbitrary rule in Π. To prove $\langle T \cap J, T\rangle \models r$ we already know that $\langle T, T\rangle \models r$ and it remains to prove that if $\langle T \cap J, T\rangle \models B \wedge N$ then $\langle T \cap J, T\rangle \models H$. So suppose that $\langle T \cap J, T\rangle \models B \wedge N$. Then $\langle T, T\rangle \models B \wedge N$ and $\langle J, J\rangle \models B$. Therefore, $\langle T, T\rangle \models H$ and there exists $p \in H$ such that $\langle T, T\rangle \models p$. Since rule $B \rightarrow p \in \Pi^\wedge$ and $\langle J, J\rangle \models B$, we get that $\langle J, J\rangle \models p$ and so $\langle T \cap J, T\rangle \models H$, as we wanted to prove. ⊠

[4] We could further simplify $\Pi^\wedge$ removing rules (5.18) and (5.20) by observing that their head predicates never occur in a positive body of $\Pi_{5.2}$. However, for the formal results, this is not essential, and would complicate the definitions.

Proposition 5.8. *For any STLP* Π, $CredFacts(\Pi) \subseteq LM(\Pi^\wedge)$.

Proof. Let $\langle \mathbf{T}, \mathbf{T} \rangle$ be any temporal equilibrium model of Π an denote by $\{L_i\}_{i \geq 0}$ the corresponding infinite sequence of ground atoms of $LM(\Pi^\wedge)$. By Theorem 5.1, we know that, for all $i \geq 0$, T_i (resp. L_i) is a stable model of $slice(\Pi, \mathbf{T}, i)$ (resp. of $slice(\Pi^\wedge, LM(\Pi^\wedge), i)$. Finally, we can apply lemma 5.4 and the fact that $slice(\Pi, \mathbf{T}, 0)^\wedge = slice(\Pi^\wedge, LM(\Pi^\wedge), 0)$ and, for $i \geq 1$,

$$slice(\Pi, \mathbf{T}, i)^\wedge \subseteq slice(\Pi^\wedge, LM(\Pi^\wedge), i).$$

⊠

Unfortunately, using $\Delta = LM(\Pi^\wedge)$ as set of derivable facts is infeasible for practical purposes, since this set contains infinite temporal facts corresponding to an "infinite run" of the transition system described by $\Pi^\wedge$. Take for instance $\Pi^\wedge_{5.2}$ for the cars scenario. Imagine a roadmap with thousands of connected cities. $LM(\Pi^\wedge)$ can tell us that, for instance, car 1 cannot reach New York in less than 316 steps, so that $\bigcirc^{315} At(1, Ny)$ is non-derivable, although $\bigcirc^{316} At(1, Ny)$ is derivable. However, in order to exploit this information for grounding, we would be forced to expand the program up to some temporal distance, and we have no hint on where to stop. Note that, on the other hand, when we represent the transition system as usual in ASP, using a bounded integer variable for representing time, then this fine-grained optimization for grounding can be applied, because the temporal path *always has a finite length*.

As a result, we will adopt a compromise solution taking a superset of $LM(\Pi^\wedge)$ extracted from a new theory, Γ_Π. This theory will collapse all the temporal facts from situation 2 on, so that all the states T_i for $i \geq 2$ will be repeated. We define Γ_Π as the result of replacing each rule $\Box(B \wedge \bigcirc B' \rightarrow \bigcirc p)$ in $\Pi^\wedge$ by the formulas:

$$B \wedge \bigcirc B' \rightarrow \bigcirc p \tag{5.21}$$
$$\bigcirc B \wedge \bigcirc^2 B' \rightarrow \bigcirc^2 p \tag{5.22}$$
$$\bigcirc^2 B \wedge \bigcirc^2 B' \rightarrow \bigcirc^2 p \tag{5.23}$$

and adding the axiom schema:

$$\bigcirc^2 \Box(p \leftrightarrow \bigcirc p) \tag{5.24}$$

for any ground atom $p \in At(D, P)$ in the signature of Π. As we can see, (5.21) and (5.22) are the first two instances of the original rule $\Box(B \wedge \bigcirc B' \rightarrow \bigcirc p)$ corresponding to situations $i = 0$ and $i = 1$. Formula (5.23), however, differs from the instance we would get for $i = 2$ since, rather than having $\bigcirc^3 B'$ and $\bigcirc^3 p$, we use $\bigcirc^2 B'$ and $\bigcirc^2 p$ respectively. This can be done because axiom (5.24) is asserting that from situation 2 on all the states are repeated.

In the cars example, for instance, rule (5.16) in $\Pi^\wedge_{5.2}$ would be transformed in $\Gamma_{\Pi_{5.2}}$ into the three rules:

$$Driveto(x, a) \rightarrow \bigcirc At(x, a)$$
$$\bigcirc Driveto(x, a) \rightarrow \bigcirc^2 At(x, a)$$
$$\bigcirc^2 Driveto(x, a) \rightarrow \bigcirc^2 At(x, a)$$

As axiom (5.24) asserts/implies that the extent we obtain for predicates at situation 2 will be repeated for the rest of future situations $i \geq 3$, it is quite easy to see that, for computing temporal stable models of Γ_Π it suffices with just considering the theory $\Gamma_\Pi \setminus \{(5.24)\}$. In fact, for any $\mathbf{M} = \langle (D, \sigma), \mathbf{T}, \mathbf{T} \rangle$, the following conditions are equivalent:

- $\mathbf{M}$ is a temporal equilibrium model of Γ_Π
- $\{\bigcirc^i p \mid p \in T_i\,,\ i = 0, 1, 2\}$ is a stable model of $\Gamma_\Pi \setminus \{(5.24)\}$ and $T_i = T_2$ for $i \geq 2$.

Doing so, we can see the remaining rules as a positive normal ASP (i.e., non-temporal) program for the propositional signature $\{p, \bigcirc p, \bigcirc^2 p \mid p \in At(D, P)\}$. This program can be fed to `DLV` to obtain a unique stable model that fixes T_0, T_1 and T_2 for $LM(\Gamma_\Pi)$. This completes our set of derivable facts since $T_i = T_2$ for $i \geq 3$ in $LM(\Gamma_\Pi)$.

Theorem 5.8. *Γ_Π has a least LTL-model, $LM(\Gamma_\Pi)$ which is a superset of $LM(\Pi^\wedge)$.*

Proof. Let $\langle \mathbf{T}, \mathbf{T} \rangle$ be the unique temporal equilibrium model of $\Pi^\wedge$ and let $\langle \mathbf{D}, \mathbf{D} \rangle$ denote the temporal interpretation defined by:

- $D_i = T_i$ if $0 \leq i \leq 1$,
- D_2 is the stable model of the positive non-disjunctive program:

$$\{\bigcirc^2 B \wedge \bigcirc^2 B' \rightarrow \bigcirc^2 p \mid \Box(B \wedge \bigcirc B' \rightarrow \bigcirc p) \in dyn(\Pi^\wedge)\} \cup slice(\Pi^\wedge, LM(\Pi^\wedge), 1)$$

- $D_i = D_2$ if ≥ 3,

It is straightforward to check that $\langle \mathbf{D}, \mathbf{D} \rangle$ is a temporal equilibrium model of Γ_Π. Notice that $T_2 \subseteq D_2$. This follows from lemma 5.4 and the facts that $\mathbf{T}^2 \setminus AT^1$ is the stable model of $slice(\Pi^\wedge, LM(\Pi^\wedge), 1)$ and D_2 is a model of this latter program.

The cases $i = 0, 1, 2$ follow from Proposition 5.8 an the fact that $T_2 \subseteq D_2$.

When $i \geq 3$, we shall prove that $\langle \mathbf{T}^i \setminus At^{i-1} \cap \mathbf{D}^i \setminus At^{i-1}, \mathbf{T}^i \setminus At^{i-1} \rangle$ is a model of $slice(\Pi^\wedge, \mathbf{T}, i)$ so by Theorem 5.1, $\mathbf{T}^i \setminus At^{i-1} \cap \mathbf{D}^i \setminus At^{i-1} = \mathbf{T}^i \setminus At^{i-1}$ and, consequently, $T_i \subseteq D_i$. So, take $\bigcirc^i B' \rightarrow \bigcirc^i H' \in slice(\Pi^\wedge, \mathbf{T}, i)$ and suppose that

$$\langle \mathbf{T}^i \setminus At^{i-1} \cap \mathbf{D}^i \setminus At^{i-1}, \mathbf{T}^i \setminus At^{i-1} \rangle \models \bigcirc^i B'$$

This fact implies that $\langle \mathbf{T}^i \setminus At^{i-1}, \mathbf{T}^i \setminus At^{i-1} \rangle \models \bigcirc^i B'$ and also that there exists a (positive normal) dynamic rule like (5.3) such that $B \subseteq T_{i-1} \subseteq D_{i-1}$. Since $\langle \mathbf{T}^i \setminus At^{i-1}, \mathbf{T}^i \setminus At^{i-1} \rangle$ is a model of $slice(\Pi^\wedge, \mathbf{T}, i)$, the only atom $p \in H'$ satisfies $\langle \mathbf{T}^i \setminus At^{i-1}, \mathbf{T}^i \setminus At^{i-1} \rangle \models \bigcirc^i p$. It only rests to show that $\langle \mathbf{D}^i, \mathbf{D}^i \rangle \models \bigcirc^i p$ or equivalently $\langle \mathbf{D}, \mathbf{D} \rangle \models \bigcirc^2 p$ (notice that $D_i = D_2$ if $i \geq 2$). Finally, we can use that the rule $\bigcirc^2 B \wedge \bigcirc^2 B' \rightarrow \bigcirc^2 p \in \Gamma_\Pi$ and also the fact that $\langle \mathbf{D}, \mathbf{D} \rangle \models \bigcirc^2 B \wedge \bigcirc^2 B'$ because $i \geq 3$ and $B' \subseteq D_i = D_2$ and $B \subseteq T_{i-1} \subseteq D_{i-1} = D_2$. ⊠

In other words $CredFacts(\Pi) \subseteq LM(\Pi^{\wedge}) \subseteq LM(\Gamma_\Pi) = \Delta$, i.e., we can use $LM(\Gamma_\Pi)$ as set of derivable facts and simplify the ground program accordingly. Note that this simplification does not mean that we first ground everything and then remove rules and literals: we simply do not generate the irrelevant ground cases.

A slight adaptation is further required for this method: as we get ground facts of the form $p, \bigcirc p$ and $\bigcirc^2 p$ we have to unfold the original STLP rules to refer to atoms in the scope of $\bigcirc^2$. For instance, given (5.10) we would first unfold it into:

$$\begin{aligned} At(x,a) \wedge Road(a,b) &\rightarrow Driveto(x,b) \vee \\ &\quad NoDriveto(x,b) && (5.25)\\ \bigcirc At(x,a) \wedge \bigcirc Road(a,b) &\rightarrow \bigcirc Driveto(x,b) \vee \\ \bigcirc NoDriveto(x,b) && (5.26)\\ \Box(\ \bigcirc^2 At(x,a) \wedge \bigcirc^2 Road(a,b) &\rightarrow \bigcirc^2 Driveto(x,b) \\ &\quad \vee \bigcirc^2 NoDriveto(x,b)\ ) && (5.27) \end{aligned}$$

and then check the possible extents for the positive bodies we get from the set of derivable facts $\Delta = LM(\Gamma_\Pi)$. For instance, for the last rule, we can make substitutions for x, a and b using the extents of $\bigcirc^2 At(x,a)$ and $\bigcirc^2 Road(a,b)$ we have in Δ. However, this still means making a join operation for both predicates. We can also use DLV for that purpose by just adding a rule that has as body, the positive body of the original temporal rule r, and as head, a new auxiliary predicate $Subst_r(x,a,b)$ referring to all variables in the rule. In the example, for rule (5.27) we would include in our DLV program:

$$\bigcirc^2 At(x,a) \wedge \bigcirc^2 Road(a,b) \rightarrow Subst_{(5.27)}(x,a,b)$$

In this way, each tuple of $Subst_r(x_1,\ldots,x_n)$ directly points out the variable substitution to be performed on the temporal rule.

5.2.3 STeLP

STeLP [19] was the first tool which is able to compute models of temporal programs under TEL semantics. This tool has been designed and developed in order to deal with splittable temporal logic programs, the subset of temporal equilibrium logic defined in Section 5.1.

Input programs of STeLP adopt the standard ASP notation for conjunction, negation and implication, so that, an initial rule like (5.1) is represented as:

$$A_{m+1}\ \texttt{v}\ldots\texttt{v}\ A_s\ \ \texttt{:-}\ \ A_1\texttt{,}\ldots\texttt{,}A_n\texttt{,}\ \texttt{not}\ A_{n+1}\texttt{,}\ldots\texttt{,}\ \texttt{not}\ A_m$$

Operator '$\bigcirc$' is represented as 'o' whereas a dynamic rule like $\Box(\alpha \rightarrow \beta)$ is written as $\beta\ \texttt{::-}\ \alpha$. In STeLP we can also use rules where atoms have variable arguments like $\texttt{p(}X_1\texttt{,}\ldots\texttt{,}X_n\texttt{)}$ and, as happens with most ASP solvers, these are understood as abbreviations of all their ground instances. A kind of *safety* condition (a special case of the safety condition of Definition 5.7) is defined for variables occurring in a rule. We will previously distinguish a family of

predicates, called *static*, that satisfy the property $\Box(\ p(\overline{X}) \leftrightarrow \bigcirc p(\overline{X})\ )$ for any tuple of elements $\overline{X}$. These predicates are declared using a list of pairs *name/arity* preceded by the keyword `static`. All built-in relational operators `=`, `!=`, `<`, `>`, `<=`, `>=` are implicitly defined as static, having their usual meaning. An initial or dynamic rule r is *safe* when:

1. Any variable X occurring in a rule $head(r) \leftarrow body(r)$ or $\Box(head(r) \leftarrow body(r))$ occurs in some positive literal in $body(r)$ for some static predicate p.
2. Initial rules of the form $head(r) \leftarrow body(r)$ where at least one static predicate occurs in $head(r)$ only contain static predicates (these are called *static rules*).

As an example, let us consider the `STeLP` input program of the Figure 5.2, which is the temporal representation of the ASP program of Figure 2.3.

```
domain item(X), object(Y).
static opp/2.     fluent at/2.      action m/1.
% Domain predicates
opp(l,r). opp(r,l).     item(w). item(g). item(c).
object(Z) :- item(Z).   object(b).
% Effect axiom for moving
o at(X,A) ::- at(X,B), m(X), opp(A,B).
% The boat is always moving
o at(b,A) ::- at(b,B), opp(A,B).
% Inertia
o at(Y,A) ::- at(Y,A), not o at(Y,B),opp(A,B).
% Action executability
::- m(X), at(b,A), at(X,B), opp(A,B).
% Unique value constraint
::- at(Y,A), at(Y,B), opp(A,B).
% Wolf eats goat
::- at(w,A), at(g,A), at(b,B), opp(A,B).
% Goat eats cabbage
::- at(g,A), at(c,A), at(b,B), opp(A,B).
 % Choice rules for action execution
a(X) ::- not m(X).
m(X) ::- not a(X).
% Non-concurrent actions
::- m(X), item(Z), m(Z), X != Z.
% Initial state
at(Y,l).
```

Figure 5.2: Wolf-goat-cabbage puzzle in `STeLP`.

Since static predicates must occur in any rule, `STeLP` allows defining global variable names with a fixed domain, in a similar way to the `lparse`[5] directive

[5] `http://www.tcs.hut.fi/Software/smodels/lparse.ps.gz`

`#domain`. For instance, the declaration `domain item(X).` means that any rule referring to variable `X` is implicitly extended by including an atom `item(X)` in its body. All predicates used in a domain declaration must be static – as a result, they will be implicitly declared as static, if not done elsewhere.

A feature that causes a difficult reading of the obtained automaton for a given STLP is that all the information is represented by formulas that occur as transition labels, whereas states are just given a meaningless name. As opposed to this, in an actions scenario, one would expect that states displayed the fluents information and transitions only contained the actions execution. To make the automaton closer to this more natural representation, we can distinguish predicates representing actions and fluents. For instance, in the previous example, we would further declare: `action m/1. fluent at/2.` STeLP uses this information so that when all the outgoing transitions from a given state share the same information for fluents, this is information is shown altogether inside the state, and removed from the arc labels. Besides, any symbol that is not an action or a fluent is not displayed (they are considered as auxiliary). As a result of these simplifications, we may obtain several transitions with the same label: if so, they are collapsed into a single one. An example of this feature is shown in Figure 5.3.

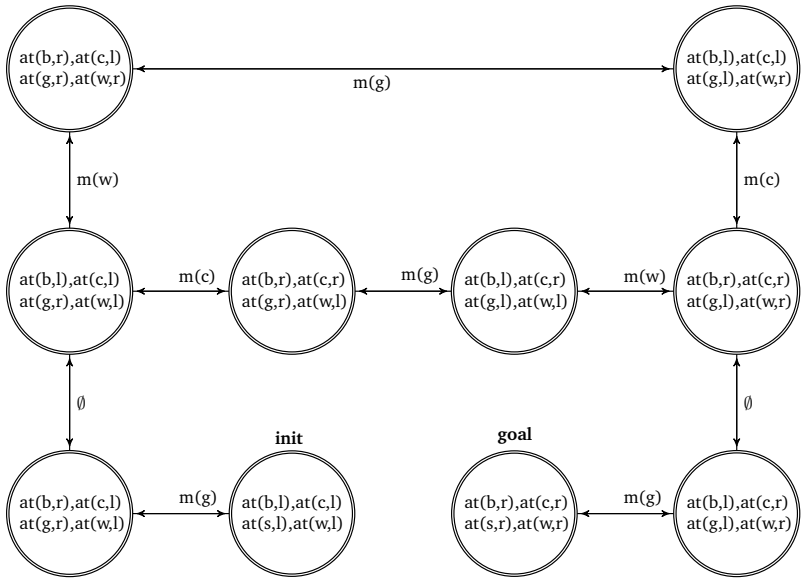


Figure 5.3: Automaton for the wolf-goat-cabbage example.

Constraints in STeLP are more general than in STLP: their body can include any arbitrary combination of propositional connectives with `o`, `always` (standing for $\Box$), `pos` (standing for $\Diamond$) and `until` (standing for $\mathcal{U}$). The empty head $\bot$ is not represented. This feature allows us to impose constraints on infinite paths. For instance if we want to restrict the set of models shown in Figure 5.3 to the ones satisfying that the three items were moved to the opposite bank of the river, we should include the theory

$$\neg\neg\Diamond\left(at(w,r) \wedge at(g,r) \wedge at(c,r)\right)$$

which is equivalent to

$$\neg\Box\neg\left(at(w,r) \wedge at(g,r) \wedge at(c,r)\right).$$

This formula can be represented in STeLP by adding the rules

```
% Goal predicate
g ::- at(w,r), at(g,r), at(c,r).
% Goal must be satisfied
:- always not g.
```

When these rules are added to the program of Figure 5.2, we obtain the Büchi automaton of Figure 5.4. It is easy to see that every accepting path reaches an state where all items were moved to the opposite bank of the river. Both a more detailed specification of the system and more examples can be found in Appendix B.

5.3 Arbitrary temporal theories

In this section, following [18], we outline the automata-based approach to determine whether a formula φ built over the propositional variables $\{p_1, \ldots, p_n\}$ has a TEL model. This method has the advantage of being applicable to any arbitrary temporal theory and also allows us to establish some complexity bounds for THT and TEL satisfiability problems. The method builds a Büchi automaton $\mathcal{B}$ over the alphabet $\Sigma = \mathcal{P}(\{p_1, \ldots, p_n\})$ such that $\mathcal{L}(\mathcal{B})$ is equal to the set of TEL consequences for φ. Based on this construction, we have developed a tool called ABSTEM[6] using, as a backend, the library SPoT[7] [31] which has been enhanced with new features such as checking different types of temporal equivalence.

5.3.1 Automata-based Computation of Temporal Equilibrium Models

Given a temporal formula φ built over a signature At, the first step of the algorithm presented in [18] consists in computing a first automaton $\mathcal{A}_1$ for capturing the THT total models, that is the set of LTL models, of φ. There exist several algorithms [120] that allow obtaining a Büchi automaton whose accepting language corresponds to the set of LTL models of φ (see [18] for a recall of a Büchi automaton construction method). Remember that temporal stable models are also LTL-models, so we need something else to reject those LTL-models that are not stable.

In a second step, we will strengthen the mapping φ'' to obtain not only THT models of φ but also to constrain them to be strictly non-total (that is

[6] Available at http://kr.irlab.org/?q=abstem
[7] Available at http://spot.lip6.fr

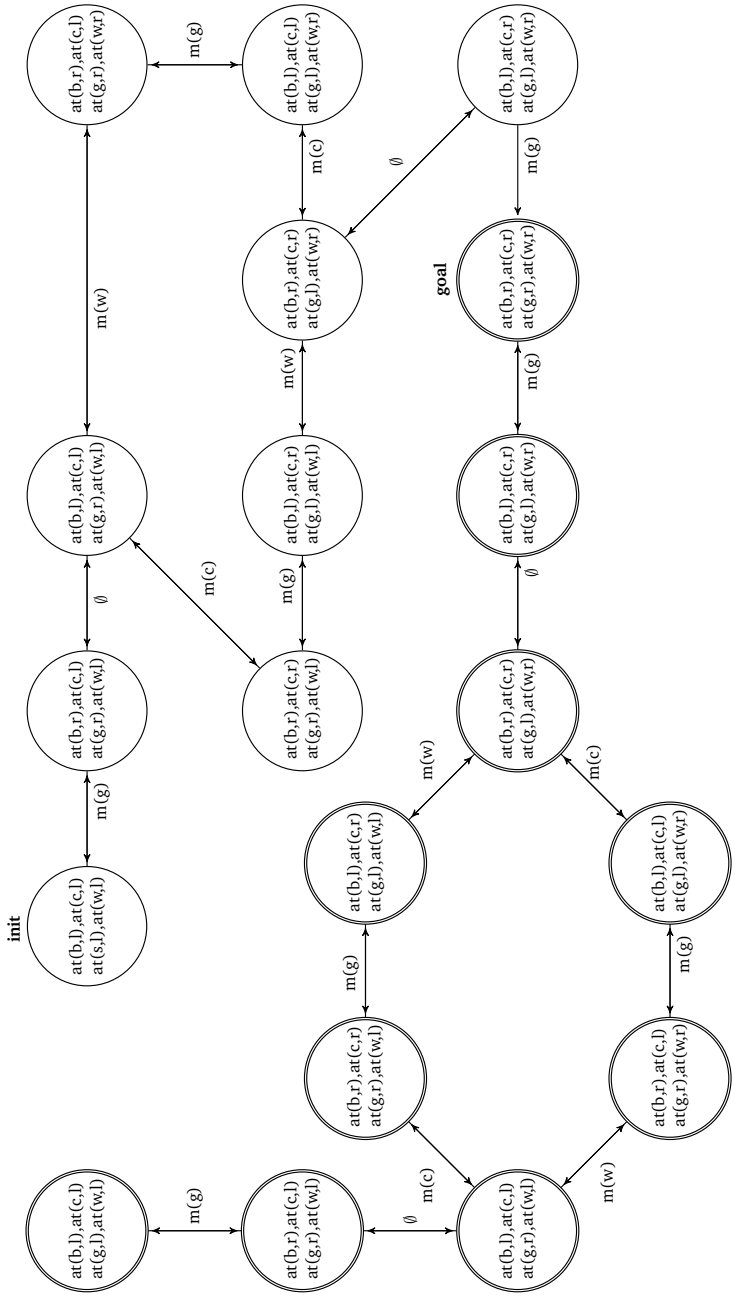


Figure 5.4: All possible solutions of the wolf-goat-cabbage example.

$\mathbf{H} < \mathbf{T}$). This operation is carried out by combining Corollary 3.2 and the axiom schema:

$$\bigvee_{p \in At} \Diamond(\neg p' \wedge p) \qquad \textbf{(Ax2)}$$

to define the expression

$$\varphi'' \stackrel{\text{def}}{=} \textbf{Ax1} \wedge (\varphi)^* \wedge \textbf{Ax2}.$$

where $(\varphi)^*$ is the extension of the star-translation from Definition 2.2.3 to the temporal case (see Section 3.1) and **Ax1** is the axiom schema, also defined in Section 3.1, which forces that $(\varphi)^*$ satisfies the property of persistence. The formula φ'' characterizes the non-total THT models of the formula φ and **Ax2** ensures that at some position j, $H_j \subset T_j$ (strict inclusion).

Lemma 5.5. *The set of LTL models for the formula φ'' corresponds to the set of non-total THT models for the temporal formula φ.* ⊠

Let $\mathcal{A}_2$ be the Büchi automaton such that $\mathcal{L}(\mathcal{A}_2) = \mathrm{Mod}(\varphi'')$, following again any construction similar to [120]. The set $\mathcal{L}(\mathcal{A}_2)$ is isomorphic to the set of non-total THT models of φ.

Note that both automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are built over different alphabets Σ and $\Sigma' = \mathcal{P}(\{p_1, \ldots, p_n, p'_1, \ldots, p'_n\})$ respectively. Let $h : \Sigma' \to \Sigma$ be a map (renaming) between the two finite alphabets such that $h(a) = a \cap \{p_1, \ldots, p_n\}$. h can be naturally extended as an homomorphism between finite words, infinite words and as a map between languages.

Similarly, given a Büchi automaton $\mathcal{A}_2 = (\Sigma', Q, Q_0, \delta, F)$, $h(\mathcal{A}_2)$ denotes the Büchi automaton $(\Sigma, Q, Q_0, \delta', F)$ such that $q \stackrel{a}{\to} q' \in \delta'$ iff there is $b \in \Sigma'$ such that $q \stackrel{b}{\to} q' \in \delta$ and $h(b) = a$. Obviously, $\mathcal{L}(h(\mathcal{A}_2)) = h(\mathcal{L}(\mathcal{A}_2))$ and $\mathcal{L}(h(\mathcal{A}_2))$ can be viewed as the set of total THT models for φ having a strictly smaller THT one.

The only remaining task is getting the intersection of total models captured by $\mathcal{L}(\mathcal{A}_1)$ with models of the complementary of $\mathcal{L}(h(\mathcal{A}_2))$, that is, those for which we do not have a strictly smaller model (or they are not models either). We have shown in Section 2.3.3 that the class of languages recognized by Büchi automata (the class of ω-regular languages) is effectively closed under union, intersection and complementation and renaming operation. Since $\mathcal{A}_1$ and $h(\mathcal{A}_2)$ are Büchi automata, one can build a Büchi automaton $\mathcal{A}'$ such that

$$\mathcal{L}(\mathcal{A}') = \Sigma^\omega \setminus \mathcal{L}(h(\mathcal{A}_2)).$$

Similarly, one can effectively build a Büchi automaton $\mathcal{B}_\varphi$ such that

$$\mathcal{L}(\mathcal{B}_\varphi) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}').$$

Complementation can be performed using the constructions in [112] or, if optimality is required, in [107].

Proposition 5.9 (from [18]). *φ has a TEL model iff*

$$\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^\omega \setminus \mathcal{L}(h(\mathcal{A}_2))) \neq \emptyset.$$

⊠

Consequently, the set of TEL models for a given formula forms an ω-regular language.

As a corollary, Cabalar and Demri, could stablish in [18] both a lower bound (PSPACE-hard) and an upper bound (EXPSPACE) for Temporal Equilibrium Logic. The former comes from the fact that LTL can be encoded in TEL an the latter from how the final automaton is computed. This result is summarised in the following theorem.

Theorem 5.9 (from [18]). *Checking whether two temporal formulae have the same TEL models is decidable in* EXPSPACE *and it is* PSPACE*-hard.* ⊠

Recently, this gap was filled in [13], fixing the complexity of TEL as EXPSPACE.

5.3.2 ABSTEM

In this section we present `ABSTEM`, a tool designed to deal with arbitrary temporal theories (see Appendix C to have an overview of the system options). In order to show a full example of `ABSTEM` input theory, we consider the non-splittable formula $\varphi = \neg p \rightarrow \Diamond q$, which means that if p cannot be proved in the initial state, then q will be satisfied once (and only one due to the model minimisation) in a future state. In TEL, we have infinite temporal equilibrium models, each one corresponding to q true in one (and only one) time point $i \geq 0$ whereas p becomes false forever. If we assume that φ is written in a file called `th1.abs`, we can obtain the two initial automata, A_φ and $A_{\varphi''}$ executing the following commands:

- `abstem -t -m -SForm -f th1.abs : ABSTEM` will produce a file called `ltl_models.png` which corresponds to the automaton A_φ shown in Figure 5.6(a).
- `abstem -t -m -SNonTot -f th1.abs :` we will get another file called `non_total.png` which corresponds to $A_{\varphi''}$ (shown in Figure 5.6(b)), and obtained from the expression

$$\begin{aligned}\varphi'' \quad = \quad & \Box(p' \rightarrow p) \wedge \Box(q' \rightarrow q) \wedge (\neg p \rightarrow \Diamond q) \wedge (\neg(p \vee p') \rightarrow \Diamond q') \\ & \wedge (\Diamond(p \wedge \neg p') \vee \Diamond(q \wedge \neg q')).\end{aligned}$$

Both automata are computed by the algorithm published in [29] and implemented in `SPoT`. Once they have been computed, the filter h is applied to $A_{\varphi''}$ by removing every auxiliary atom from the transitions of $A_{\varphi''}$. Then $h(A_{\varphi''})$, shown in Figure 5.6(c), is complemented by using Safra's complementation [107] (the result is shown in Figure 5.6(d)) and finally, the common language accepted by both A_φ and $\overline{h(A_{\varphi''})}$ is obtained by means of the

Automata	File	Command
A_φ	`ltl_models.png`	`abstem -t -m -SForm -f th1.abs`
$A_{\varphi''}$	`non_total.png`	`abstem -t -m -SNonTot -f th1.abs`
$h(A_{\varphi''})$	`filtered.png`	`abstem -t -m -SFiltered -f th1.abs`
$\overline{h(A_{\varphi''})}$	`complementary.png`	`abstem -t -m -SComp -f th1.abs`
$A_\varphi \otimes \overline{h(A_{\varphi''})}$	`tem.png`	`abstem -t -m -f th1.abs`

Table 5.1: Commands that are necessary to compute every intermediate automaton in ABSTEM.

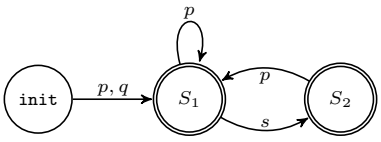
lazy product of both automata. Figure 5.6(e) shows the final automaton, $A_\varphi \otimes \overline{h(A_{\varphi''})}$, whose accepting language corresponds to the set of temporal stable models of φ. The sequence of commands to obtain all intermediate automata are outlined in Table 5.1. For a more detailed description of ABSTEM usage see Appendix C.

Unfortunately, because of TEL's complexity, intermediate automata become more unreadable and complex as the number of atoms of the theory increases. For instance, let us consider the theory Γ shown in 5.5(a). While rules (5.28)-(5.30) can be analysed by STeLP, the (non-splittable) rule (5.31) could not be represented neither in ASP nor STeLP. Intuitively (5.31) allows deriving q if p occurs "infinitely often" (something that follows from (5.28)-(5.30).

If we use ABSTEM to compute the set of temporal equilibrium models of such theory, we will obtain the Büchi automaton of Figure 5.5(b) where, as we can see, atom q is made true at the initial state, pointing out that p occurs infinitely often due to the other rules. However, the intermediate automata are more complex. For instance A_φ consists of 5 states and 13 transitions, $A_{\varphi''}$ has 19 states and 592 transitions and, finally, $\overline{h(A_{\varphi''})}$ consists of 19 states and 123 transitions.

$$\neg r \rightarrow p \quad (5.28)$$
$$\Box(p \rightarrow \bigcirc p \vee \bigcirc s) \quad (5.29)$$
$$\Box(s \wedge \neg p \rightarrow \bigcirc p) \quad (5.30)$$
$$(\Box\Diamond p) \rightarrow q \quad (5.31)$$

(a) Γ_1



(b) TEM's of Γ_1

Figure 5.5: Example of a non-splittable theory Γ_1 and its set of temporal equilibrium models.

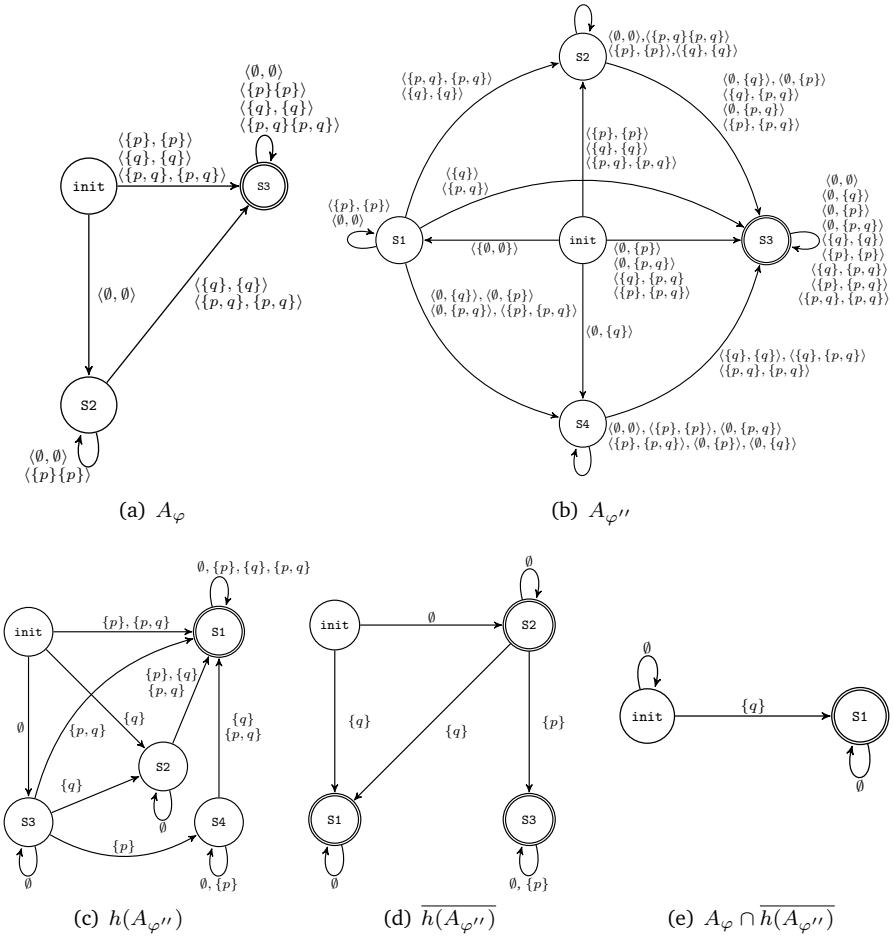


Figure 5.6: Intermediate automata generated in the example $\varphi = \neg p \rightarrow \Diamond q$.

Chapter 6

Temporal Strong Equivalence

As mentioned in Section 2.2.4, *strong equivalence* is a very important property if we want to check whether two theories have the same behaviour in a non-monotonic formalism. In the case of TEL, this property was firstly considered in [2]. In that paper, authors showed that THT-equivalence is obviously a sufficient condition for strong equivalence in TEL and, furthermore they implemented a prototype checker and used it on some examples. However, since [2], the question whether THT-equivalence was also necessary or not remained unanswered. This raised doubts on the adequacy of THT as a basis for TEL and had also some practical negative consequences. In particular, when two theories Γ_1, Γ_2 were not THT-equivalent, the checker in [2] could not answer anything about strong equivalence, while one would expect to be provided with a negative answer plus some context Γ that made them behave differently.

In this chapter we provide a proof[1], inspired in the one developed in [79] for the case of first-order non-monotonic theories, of such a necessary condition and, moreover, we provide an algorithm for checking strong equivalence by applying several ω-automata transformations that have been also used in [18] for computing the set of temporal equilibrium models of arbitrary theories.

Apart from these theoretical results, we also extend the functionalities of `ABSTEM` with the possibility of checking three different types of equivalence such as LTL-equivalence, weak equivalence (coincidence in temporal stable models) and THT-equivalence which, being a necessary and sufficient condition corresponds to strong equivalence. When a negative answer for strong equivalence is obtained, the tool also suggests a context formula Γ that makes Γ_1 and Γ_2 behave differently and generates a Büchi automaton which captures either one or all temporal stable models in the difference.

[1] The main results of this chapter have been published in [20]

6.1 Temporal Strong Equivalence

Concerning to temporal scenarios, checking whether two arbitrary temporal theories[2], Γ_1 and Γ_2, are strongly equivalent introduces, in fact, a temporal dimension in the study of strong equivalence for ASP, since it means showing that Γ_1 and Γ_2 have the same behaviour not only under any hypothetical context, but also for narratives with unlimited time. Of course, resorting to temporal logic is not always necessary. For instance, when we just deal with transition systems, a straightforward possibility[3] is to restrict the study to transitions between two consecutive states, say 0 and 1, using the non-temporal approach to check strong equivalence (that is, HT-equivalence). However, when non-splittable formulas are involved, that technique is not possible any more. As an example of the kind of difficulties we find when we move to non-splittable theories, consider the theory Γ_2:

$$\Box(p \wedge q \rightarrow \bigcirc q) \tag{6.1}$$
$$\Box(\neg p \wedge \bigcirc p \rightarrow \bigcirc\bigcirc q) \tag{6.2}$$
$$p \rightarrow \bot \tag{6.3}$$
$$\Box(p \wedge q \rightarrow \bot) \tag{6.4}$$

where (6.2) is non-splittable, since it checks atoms at three different situations. As the rule is preceded by $\Box$, if we wanted to make a static analysis, we should take transitions for states $0, 1, 2$ but also for $1, 2, 3$. This should be carefully chosen by hand and, in any case, it is difficult to conclude, for instance, that in the theory above, the first two formulas (6.1) and (6.2) can be safely replaced by:

$$\Box(p \rightarrow \bigcirc q) \tag{6.5}$$

as stated below.

Proposition 6.1. *Theory* $\Gamma_3 =$ (6.3)-(6.5) *is strongly equivalent to* $\Gamma_2 =$(6.1)-(6.4).

Proof. Using Theorem 6.1, we will show that Γ_2 and Γ_3 are THT-equivalent. We begin recalling a pair of LTL theorems that are also valid in THT:

$$\Box(\alpha \wedge \beta) \leftrightarrow \Box\alpha \wedge \Box\beta \tag{6.6}$$
$$\Box\alpha \leftrightarrow \alpha \wedge \Box(\alpha \rightarrow \bigcirc\alpha) \tag{6.7}$$
$$\bigcirc(\alpha \rightarrow \beta) \leftrightarrow (\bigcirc\alpha \rightarrow \bigcirc\beta) \tag{6.8}$$

We will also use the HT-valid equivalences that allow unnesting implications:

$$\alpha \rightarrow (\beta \rightarrow \gamma) \leftrightarrow \alpha \wedge \beta \rightarrow \gamma \tag{6.9}$$
$$(\alpha \rightarrow \beta) \rightarrow \gamma \leftrightarrow (\neg\alpha \rightarrow \gamma) \wedge (\beta \rightarrow \gamma) \wedge (\alpha \vee \neg\beta \vee \gamma) \tag{6.10}$$

[2]For simplicity, we assume finite theories and we indistinctly represent them as the conjunction of their formulas.

[3]Suggested by Joohyung Lee during conference LPNMR'11.

(6.7) is the induction schema: applied to (6.5) we get

$$\begin{array}{rl} & \Box(p \rightarrow \bigcirc q) \\ \leftrightarrow & (p \rightarrow \bigcirc q) \wedge \Box((p \rightarrow \bigcirc q) \rightarrow \bigcirc(p \rightarrow \bigcirc q)) \end{array}$$

By (6.8) we get:

$$\leftrightarrow \quad (p \rightarrow \bigcirc q) \wedge \Box((p \rightarrow \bigcirc q) \rightarrow (\bigcirc p \rightarrow \bigcirc \bigcirc q))$$

We unfold the nested implications using (6.9) and (6.10) and use (6.6) to distribute $\Box$ on conjunction afterwards, obtaining:

$$\begin{array}{rl} \leftrightarrow & (p \rightarrow \bigcirc q) \wedge \Box(\neg p \wedge \bigcirc p \rightarrow \bigcirc \bigcirc q) \\ & \wedge \Box(\bigcirc q \wedge \bigcirc p \rightarrow \bigcirc \bigcirc q) \\ & \wedge \Box(\bigcirc p \rightarrow p \vee \neg \bigcirc q \vee \bigcirc \bigcirc q) \end{array} \tag{6.11}$$

So we concluded that (6.5) is equivalent to (6.11). Now, from (6.4) it is easy to see that $\Box(p \rightarrow \neg q)$ and in particular, $\Box(\bigcirc p \rightarrow \neg \bigcirc q)$ something that implies the last conjunct of (6.11). On the other hand, as $\neg p$ follows from (6.3), we conclude that the following implications also hold:

$$p \rightarrow \bigcirc q \qquad\qquad q \wedge p \rightarrow \bigcirc q$$

so that we can also remove the first conjunct in (6.11) whereas the third one can be replaced by $\Box(q \wedge p \rightarrow \bigcirc q)$. To sum up, when constraints (6.3) and (6.4) are present, (6.11) is eventually equivalent to:

$$\Box(\neg p \wedge \bigcirc p \rightarrow \bigcirc \bigcirc q) \wedge \Box(q \wedge p \rightarrow \bigcirc q)$$

which is the conjunction of (6.1) and (6.2). ⊠

In fact, if we check these two theories using `ABSTEM` it just provides a positive answer and no further explanation is required. As we can see, the proof for Proposition 6.1 relies on the fact that Γ_2 and Γ_3 are THT-equivalent and, as we explained before, this is trivially a sufficient condition for strong equivalence (see [2]).

Theorem 6.1 (sufficient condition)**.** *If two temporal formulas α and β are THT-equivalent then they are strongly equivalent in TEL.*

Proof. The proof is straightforward. If α and β are satisfied by the same THT-interpretations then $\alpha \wedge \gamma$ and $\beta \wedge \gamma$ (for every formula γ) are also satisfied by the same THT-interpretations. But then, selecting among them the temporal equilibrium models yields the same effect on both. ⊠

Until now, however, we missed the other direction, namely, that the property of THT-equivalence is also a necessary condition for strong equivalence. In the rest of this section, we prove this result adapting the main proof in [79]. Before introducing the main theorem we will first introduce some auxiliary results that will be used for the main proof and for implementation purposes. We begin defining the following axiom:

$$\gamma_0 \quad \stackrel{\text{def}}{=} \quad \bigwedge_{p \in At} \Box(p \vee \neg p)$$

The effect of adding this axiom to a theory is restricting to total models, as stated below:

Proposition 6.2. *Let $\langle \mathbf{H}, \mathbf{T}\rangle$ be a THT interpretation for signature At. If $\langle \mathbf{H},\mathbf{T}\rangle \models \gamma_0$ then $\mathbf{H} = \mathbf{T}$.*

Proof. We have to prove that $\forall i \in \mathbb{N}$, $H_i = T_i$. Since $H_i \subseteq T_i$ from $\mathbf{H} \leq \mathbf{T}$, we just need to prove $T_i \subseteq H_i$. Take some $p \in T_i$. Obviously, $\langle \mathbf{H}, \mathbf{T}\rangle, i \not\models \neg p$. But from $\langle \mathbf{H}, \mathbf{T}\rangle \models \Box(p \vee \neg p)$ we conclude $\langle \mathbf{H}, \mathbf{T}\rangle, i \models p \vee \neg p$ and so $\langle \mathbf{H}, \mathbf{T}\rangle, i \models p$ that is $p \in H_i$. ⊠

Corollary 6.1. *For any formula α for signature At, the LTL-models of $\alpha \wedge \gamma_0$ coincide with its TS-models.*

Proof. By Observation 3.1, any TS-model of $\alpha \wedge \gamma_0$ is an LTL-model too. For the other direction, by Proposition 6.2, any THT-model of $\alpha \wedge \gamma_0$ is a total model. Since $\alpha \wedge \gamma_0$ has no non-total models, any model $\langle \mathbf{T}, \mathbf{T}\rangle$ is a temporal equilibrium model and so $\mathbf{T}$ is a TS-model. ⊠

Lemma 6.1. *Let α and β be two LTL-equivalent formulas and let $\gamma = (\beta \rightarrow \gamma_0)$. Then, the following conditions are equivalent:*

(i) *There exists some $\mathbf{H} < \mathbf{T}$ such that $\langle \mathbf{H}, \mathbf{T}\rangle \not\models \alpha \rightarrow \beta$;*

(ii) *$\mathbf{T}$ is TS-model of $\beta \wedge \gamma$ but not TS-model of $\alpha \wedge \gamma$.*

Proof. For (i) $\Rightarrow$ (ii), suppose (i) holds. As α and β are LTL-equivalent, $\langle \mathbf{H}, \mathbf{T}\rangle \not\models \alpha \rightarrow \beta$ amounts to $\langle \mathbf{H}, \mathbf{T}\rangle \models \alpha$ and $\langle \mathbf{H}, \mathbf{T}\rangle \not\models \beta$. Then, it is easy to see that $\langle \mathbf{H}, \mathbf{T}\rangle \models \alpha \wedge \gamma$ because $\langle \mathbf{H}, \mathbf{T}\rangle \not\models \beta$ which is the antecedent of γ. From Proposition 3.1 (persistence) $\langle \mathbf{T}, \mathbf{T}\rangle \models \alpha \wedge \gamma$ and since α and β have the same total models, $\langle \mathbf{T}, \mathbf{T}\rangle \models \beta \wedge \gamma$ that is, $\mathbf{T} \models \beta \wedge \gamma$ in LTL. But now, as $\beta \wedge (\beta \rightarrow \gamma_0)$ is LTL-equivalent to $\beta \wedge \gamma_0$, by Corollary 6.1, the LTL models of this formula are its TS-models. In particular $\mathbf{T}$ is a TS-model of $\beta \wedge \gamma$. But $\mathbf{T}$ cannot be TS-model of $\alpha \wedge \gamma$ because we had that $\langle \mathbf{H}, \mathbf{T}\rangle$ was a model and $\mathbf{H} < \mathbf{T}$.

For (ii) $\Rightarrow$ (i), suppose (ii) is true. Then, by Observation 3.1, $\mathbf{T}$ is LTL-model of $\beta \wedge \gamma$, and thus, it is LTL-model of $\alpha \wedge \gamma$ too, since α and β are LTL-equivalent. But as $\mathbf{T}$ is not TS-model of $\alpha \wedge \gamma$, this means there exists some $\mathbf{H} < \mathbf{T}$ such that $\langle \mathbf{H}, \mathbf{T}\rangle \models \alpha \wedge \gamma$ and so $\langle \mathbf{H}, \mathbf{T}\rangle \models \alpha$. On the other hand, $\langle \mathbf{H}, \mathbf{T}\rangle \not\models \beta$ because, otherwise, it would satify $\beta \wedge \gamma$ and, as $\langle \mathbf{H}, \mathbf{T}\rangle$ is non-total, $\mathbf{T}$ could not be TS-model of $\beta \wedge \gamma$. As a result, $\langle \mathbf{H}, \mathbf{T}\rangle \not\models \alpha \rightarrow \beta$. ⊠

Theorem 6.2 (necessary condition)**.** *If two temporal formulas α and β are strongly equivalent in TEL then they are THT-equivalent.*

Proof. The proof follows [79] although, for implementation purposes, it is more convenient here to prove the contraposition of the result, that is: if α and β are not THT-equivalent then there is some context formula γ for which $\alpha \wedge \gamma$ and $\beta \wedge \gamma$ have different TS-models.

Assume first that α and β have different total models, i.e., different LTL-models. Then, the LTL-models of $\alpha \wedge \gamma_0$ and $\beta \wedge \gamma_0$ also differ (because γ_0 is an

LTL tautology). But by Corollary 6.1, LTL-models of these theories are exactly their TS-models, which therefore, also differ.

Suppose now that α and β are LTL-equivalent but not THT-equivalent. Then, there is some THT-countermodel $\langle \mathbf{H}, \mathbf{T} \rangle$ of either $(\alpha \to \beta)$ or $(\beta \to \alpha)$, and given LTL-equivalence of α and β, the countermodel is non-total, $\mathbf{H} < \mathbf{T}$. Without loss of generality, assume $\langle \mathbf{H}, \mathbf{T} \rangle \not\models \alpha \to \beta$. By Lemma 6.1, taking the formula $\gamma = (\beta \to \gamma_0)$, we get that $\mathbf{T}$ is TS-model of $\beta \wedge \gamma$ but not TS-model of $\alpha \wedge \gamma$. ⊠

Corollary 6.2. *Let α and β be two LTL-equivalent formulas and let $\gamma = (\beta \to \gamma_0)$. Then, any TS-model of $\alpha \wedge \gamma$ is a TS-model of $\beta \wedge \gamma$.*

Proof. Suppose $\mathbf{T}$ is a TS-model of $\alpha \wedge \gamma$. By Observation 3.1, $\mathbf{T}$ is an LTL model of $\alpha \wedge \gamma$ too, and so, it is an LTL model of $\beta \wedge \gamma$, because α and β are LTL-equivalent. But, as discussed in the proof of Theorem 6.2, any LTL-model of $\beta \wedge \gamma$ is also a TS-model of that theory. ⊠

6.2 Implementation and a practical example

In this section we present a procedure for checking strong equivalence shown in Algorithm 2. It takes two arbitrary propositional temporal formulas α and β and returns either **true**, if they are strongly equivalent, or a triple with a formula γ and two automata A_1, A_2 otherwise. The meaning of this information is that A_1 captures TS-models of $\alpha \wedge \gamma$ that are not TS-models of $\beta \wedge \gamma$ and, analogously, A_2 captures TS-models of $\beta \wedge \gamma$ that are not TS-models of $\alpha \wedge \gamma$. The procedure uses several auxiliary routines that are explained below:

- **ltl_to_Büchi(φ)**: this function uses a `SPoT` function to obtain a Büchi automaton from an LTL-formula φ.
- **one_path(A)**: this function returns a single path from an automaton A
- $h(A)$: it is the result of filtering out primed atoms from automaton A (as we explained in Section 5.3)

This algorithm works as follows. The first three **if**'s in Algorithm 2 check LTL-equivalence. These steps are trivial, except that when option 'compute_one' is selected and the theories are not LTL-equivalent, only one path of the first non-empty automaton is returned. As stated in the proof of the main theorem, when LTL-equivalence fails, we take γ_0 as context formula.

If α and β are LTL-equivalent, the algorithm proceeds to find all non-total $\mathbf{H} < \mathbf{T}$ countermodels of $\alpha \to \beta$. These non-total countermodels are captured by an automaton A for the formula $\neg(\alpha \to \beta)^* \wedge (\mathbf{Ax1}) \wedge (\mathbf{Ax2})$ (firstly used in [18] for computing non-total THT-models) where $(\cdot)^*$ is the translation defined in Section 3.1.2 and both $(\mathbf{Ax1})$ and $(\mathbf{Ax2})$ are defined in Section 5.3.

If the language accepted by this automaton is not empty, by Lemma 6.1, the $\mathbf{T}$ components of these non-total countermodels are precisely the TS-models of $\beta \wedge \gamma$ that are not TS-models of $\alpha \wedge \gamma$, for $\gamma = (\beta \to \gamma_0)$. To obtain those $\mathbf{T}$ components we compute $A_2 := h(A)$ that, as explained in Section 5.3, filters

out the auxiliary atoms representing truth in **H**. The algorithm then returns $\langle \gamma, \emptyset, A_2 \rangle$ since, by Corollary 6.2, there are no TS-models of $\alpha \wedge \gamma$ that are not TS-models of $\beta \wedge \gamma$. If, on the contrary, the language of A is empty, we proceed in the analogous way for the other direction $\beta \rightarrow \alpha$. Finally, if in both cases we get an empty automaton, then this means that any non-total interpretation is a model of $\alpha \leftrightarrow \beta$ something that, together with the LTL-equivalence of α and β, means that the two formulas are THT-equivalent, that is, strongly equivalent.

Algorithm 2: StrongEquivalenceTest(α, β)

Require: Two propositional temporal formulas α, β.
If option 'compute_one' is set, it returns just one TS-model when a difference is found.

Ensure: If α and β are THT-equivalent, it returns **true**. Otherwise, it returns a triple $\langle \gamma, A_1, A_2 \rangle$ where γ is a formula and A_1, A_2 are two automata such that:
A_1 captures TS-models($\alpha \wedge \gamma$) $\setminus$ TS-models($\beta \wedge \gamma$)
A_2 captures TS-models($\beta \wedge \gamma$) $\setminus$ TS-models($\alpha \wedge \gamma$).

A_1 := ltl_to_Büchi($\alpha \wedge \neg\beta$)
if compute_one **and** $A_1 \neq \emptyset$ **then**
  return $\langle \gamma_0, \text{one\_path}(A_1), \emptyset \rangle$
end if
A_2 := ltl_to_Büchi($\beta \wedge \neg\alpha$)
if compute_one **and** $A_2 \neq \emptyset$ **then**
  return $\langle \gamma_0, \emptyset, \text{one\_path}(A_2) \rangle$
end if
if $A_1 \neq \emptyset$ **or** $A_2 \neq \emptyset$ **then**
  return $\langle \gamma_0, A_1, A_2 \rangle$
end if
A=ltl_to_Büchi($\neg(\alpha \rightarrow \beta)^* \wedge$ (**Ax1**) $\wedge$ (**Ax2**))
if $A \neq \emptyset$ **then**
  $A_2 := h(A)$
  if compute_one **then**
    A_2 := one_path(A_2)
  end if
  return $\langle (\beta \rightarrow \gamma_0), \emptyset, A_2 \rangle$
end if
A=ltl_to_Büchi($\neg(\beta \rightarrow \alpha)^* \wedge$ (**Ax1**) $\wedge$ (**Ax2**))
if $A \neq \emptyset$ **then**
  $A_1 := h(A)$
  if compute_one **then**
    A_1 := one_path(A_1)
  end if
  return $\langle (\alpha \rightarrow \gamma_0), A_1, \emptyset \rangle$
end if
return true

$$p \vee q \quad (6.12)$$
$$\Box(p \wedge \neg \bigcirc q \rightarrow \bigcirc p) \quad (6.13)$$
$$\Box(q \wedge \neg \bigcirc p \rightarrow \bigcirc q) \quad (6.14)$$
$$\Box(p \vee \neg p) \quad (6.15)$$

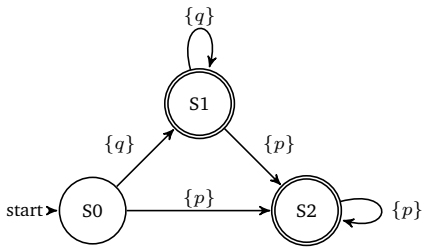


Figure 6.1: TS-models of theory (6.12)-(6.15).

Finally, to show how ABSTEM can be used to check different types of equivalence between two arbitrary theories, let us consider the temporal theory (6.12)-(6.15) of Figure 6.1, whose models coincide with sequences of states of the forms $\{q\}^*\{p\}^\omega$ or $\{q\}^\omega$. Notice how p and q are never true simultaneously, whereas once p becomes true, it remains true forever. Suppose that we add the rule

$$\Box(\neg p \rightarrow q) \quad (6.16)$$

trying to capture the idea that, when no information on p is available, q becomes true. This new rule is actually a new default for q that interacts with inertia rules (6.13),(6.14) destroying somehow their effect. Let β_1 be this extended theory, (6.12)-(6.15) plus (6.16). We can use ABSTEM to check the TS-models of β_1 (stored in file `beta1.abs`) as follows:

```
abstem -t -m -f beta1.abs
```

and we obtain the automaton in Figure 6.2(a) which corresponds to arbitrary sequences formed with states $\{p\}$ and $\{q\}$ – note the difference with respect to Figure 6.1 where p remained true after becoming true. This set of TS-models actually coincides with what one would expect from a formula of the form $\Box(p \vee q)$ since, as happens in ASP, truth minimality converts the disjunction into an exclusive or. Let us call $\alpha_1 = \Box(p \vee q)$. We check[4] whether α_1 and β_1 have the same TS-models:

```
abstem -w -m -f alpha1.abs -f beta1.abs
```

and we obtain a positive answer. Furthermore, by a quick inspection on β_1 we can foresee that it is actually LTL-equivalent to α_1. First, in LTL (6.16) is equivalent to α_1, and (6.15) is just a tautology. The other three rules, (6.12)-(6.14) can be rewritten in LTL as $(p \vee q) \wedge \Box((p \vee q) \rightarrow \bigcirc(p \vee q))$ and this is equivalent to $\Box(p \vee q)$ too (it corresponds to the induction schema for $p \vee q$). We can use ABSTEM to confirm LTL-equivalence as follows:

```
abstem -l -m -f alpha1.abs -f beta1.abs
```

and we obtain again a positive answer. However, α_1 and β_1 are not THT-equivalent and so, they are not strongly equivalent. The strong equivalence checking in ABSTEM is done as follows:

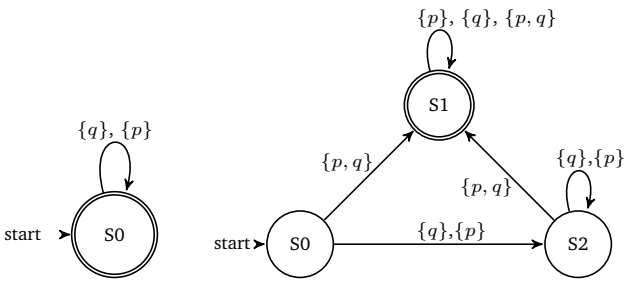
[4]See Appendix C to see all command-line options of ABSTEM

```
abstem -s -m -f alpha1.abs -f beta1.abs
```

and the answer displayed this time is negative, containing this information:

```
Not strongly equivalent:
adding the context
( (p | q) & G(p | !p)
& G((p & !Xq)->Xp) & G((q & !Xp)->Xq)
& G(!p -> q))->
  (G(p | !p) & G(q | !q))
File seq_differences_0: (p & q){(q)}w
TS-model of beta1.abs but not alpha1.abs
```

The context formula just corresponds to $\gamma_1 = \beta_1 \rightarrow \gamma_0$ in ABSTEM syntax. The output already contains a path of states $\{p,q\}\{q\}^\omega$ that is a TS-model of $\beta_1 \wedge \gamma_1$ but not of $\alpha_1 \wedge \gamma_1$. The automaton in file `seq_differences_0` is shown in Figure 6.2(b) and captures all the TS-models of $\beta_1 \wedge \gamma_1$ that are not TS-models of $\alpha_1 \wedge \gamma_1$. Note that this contains all TS-models that differ since, by Corollary 6.2, TS-models of $\alpha_1 \wedge \gamma_1$ are also TS-models of $\beta_1 \wedge \gamma_1$.



(a) TS-models of α_1 and β_1.

(b) TS-models of $\beta_1 \wedge \gamma_1$ not of $\alpha_1 \wedge \gamma_1$.

Figure 6.2: Temporal stable models related to α_1 and β_1.

Chapter 7

Related Work

In this chapter we study the relation to other approaches that have combined temporal modal operators and non-monotonic reasoning or logic programming (LP) semantics. We begin with a pair of LP approaches that cope with modal operators, TEMPLOG and Temporal Answer Sets (TAS), and move later to consider other works closer to Reasoning About Actions and Planning.

7.1 Relation between TEL and TEMPLOG

In this section, we will show that TEMPLOG is subsumed by TEL, that is, the latter can be used as a generalization of the former for syntax-independent disjunctive programs with classical and default negation. The syntax of TEMPLOG is defined next.

Definition 7.1 (TEMPLOG syntax)**.** *Let P denote an atom and N a next-atom, that is, a formula of the form $\bigcirc^i P$ for some $i > 0$. Let ϵ denote the empty expression. Then, a TEMPLOG program consists of a set of initial, permanent and goal clauses grammatically defined as:*

Non-empty Body:	B	$::= P \parallel B_1, B_2 \parallel \bigcirc B \parallel \Diamond B$
Body:	D	$::= \epsilon \parallel B$
Initial clause:	IC	$::= N \leftarrow D \parallel \Box N \leftarrow D$
Permanent clause:	PC	$::= \Box(N \leftarrow D)$
Goal clause:	G	$::= \leftarrow D$

⊠

Any TEMPLOG program can be equivalently translated into a (possibly infinite) definite program Π^* containing a set of clauses in which the only temporal operator occurs in the shape of next-atoms $\bigcirc^i P$. Π^* is obtained by replacing every occurrence of $\Diamond B$ an expression involving the next operator. We show the informal definition given by Baudinet below.

Definition 7.2 (temporally ground instance of a body (from [11]))**.** *Let B be a TEMPLOG body. A* temporally ground instance *(TGI) of B is a TG body obtained from B by replacing every occurrence of $\Diamond$ by a finite number of $\bigcirc$'s.* ⊠

For instance, let us consider the formula

$$\Diamond\left(p, \bigcirc\Diamond q\right).$$

Its corresponding set of ground instances would correspond to substitutions of the form

$$\bigcirc^{i}\left(p, \bigcirc^{j+1} q\right)$$

with both, i and j ranging on $\mathbb{N}$. In [11], this definition was actually used by assuming that the $\bigcirc$ operator were further pushed inside the expressions until it only formed temporal atoms like $\bigcirc^{i} P$. For instance, the temporal ground body

$$\bigcirc^{4}\left(p, \bigcirc^{9} q\right)$$

would correspond to the expression

$$\bigcirc^{4} p, \bigcirc^{13} q.$$

It is easy to see that, doing that, we remove all nested operators. For facilitating the correspondence proof, we develop below an equivalent formalisation of Definition 7.2 that both defines TGI in an inductive way and pushes the operator $\bigcirc$ as explained above.

Definition 7.3 (TGI operator)**.** *Let B and i be a temporal body and an integer such that $i \geq 0$ respectively. $TGI(B, i)$ is recursively defined as follows:*

- $TGI(\epsilon, i) \stackrel{\text{def}}{=} \emptyset$
- $TGI(P, i) \stackrel{\text{def}}{=} \bigcirc^{i} P$*, with P a propositional atom.*
- $TGI((B_1, B_2), i) \stackrel{\text{def}}{=} \{\left(B_1^{'}, B_2^{'}\right) \mid B_1^{'} \in TGI(B_1, i)$ *and* $B_2^{'} \in TGI(B_2, i)\}$.
- $TGI(\bigcirc B, i) \stackrel{\text{def}}{=} TGI(B, i+1)$.
- $TGI(\Diamond B, i) \stackrel{\text{def}}{=} \bigcup_{k=0}^{\infty} TGI(B, i+k)$.

⊠

It can be proved that this definition is equivalent to the translation $\| \, \|_k$, from THT into infinitary formulas, defined in Section 3.4.

Proposition 7.1. *Let $M^{\infty} = \langle H^{\infty}, T^{\infty} \rangle$ be an infinitary HT interpretation (see an overview of Infinitary Equilibrium Logic in Section 2.2.6) and B a TEMPLOG body. It holds that*

$$M^{\infty} \models \{\alpha \mid \alpha \in TGI(B, i)\}^{\vee} \quad \textit{iff} \quad M^{\infty} \models \|B\|_i$$

where we assume that ',' operator in each body in $TGI(B, i,)$ is interpreted as '$\wedge$' in infinitary HT.

Proof. We proceed by structural induction. The proof for the base case comes from the definition of the translation.

$$M^\infty \models TGI(\epsilon, i)^\vee \Leftrightarrow M^\infty \models \emptyset^\vee$$
$$\Leftrightarrow M^\infty \models \|\bot\|_i \quad \text{(Def. 3.6)}$$

$$M^\infty \models TGI(P, i)^\vee \Leftrightarrow M^\infty \models \bigcirc^i P$$
$$\Leftrightarrow M^\infty \models \|P\|_i \quad \text{(Def. 3.6)}$$

Now we continue with the inductive step. For the case of conjunction, if we apply Definition 7.3, we get

$$M^\infty \models TGI(B_1, B_2, i)^\vee \Leftrightarrow \begin{array}{l} M^\infty \models \{B_1^{'}, B_2^{'} \mid B_1^{'} \in TGI(B_1, i)^\vee \text{ and} \\ B_2^{'} \in TGI(B_2, i)^\vee\}^\vee \end{array}$$

which, because of the distributivity property of infinitary HT, it corresponds to

$$M^\infty \models TGI(B_1, B_2, i)^\vee \Leftrightarrow M^\infty \models TGI(B_1, i)^\vee \text{ and } M^\infty \models TGI(B_2, i)^\vee$$

and the chain of equivalences would continue as follows:

$$M^\infty \models TGI(B_1, i)^\vee \text{ and } M^\infty \models TGI(B_2, i)^\vee \Leftrightarrow \begin{array}{l} M^\infty \models \|B_1\|_i \\ \text{and } M^\infty \models \|B_2\|_i \end{array} \text{(Ind.)}$$

$$\Leftrightarrow M^\infty \models \|B_1, B_2\|_i \quad \text{(Def. 3.6)}$$

$$M^\infty \models TGI(\bigcirc B, i)^\vee \Leftrightarrow M^\infty \models TGI(B, i+1)^\vee \quad \text{(Def. 7.3)}$$
$$\Leftrightarrow M^\infty \models \|B\|_{i+1} \quad \text{(Ind.)}$$
$$\Leftrightarrow M^\infty \models \| \bigcirc B\|_i \quad \text{(Def. 3.6)}$$

$$M^\infty \models TGI(\Diamond B, i)^\vee \Leftrightarrow M^\infty \models \{B^{'} \mid B^{'} \in TGI(B, i+k)\}^\vee \quad \text{(Def. 7.3)}$$

$$\Leftrightarrow \begin{array}{l} \exists\, k \geq 0 \text{ and } B^{'} \in TGI(B, i+k)^\vee \text{ s.t.} \\ M^\infty \models B^{'} \end{array} \quad \text{(Def. 2.25)}$$

$$\Leftrightarrow M^\infty \models TGI(B, i+k)^\vee, \text{ with } k \geq 0 \quad \text{(Def. 2.25)}$$
$$\Leftrightarrow M^\infty \models \|B\|_{i+k} \quad \text{(Ind.)}$$
$$\Leftrightarrow M^\infty \models \|\Diamond B\|_i \quad \text{(Def. 3.6)}$$

⊠

The program Π^* is said to be temporal ground, concept defined in terms Definition 7.2. We adapt such definition in order to use TGI operator.

Definition 7.4 (temporal ground instance of a clause adapted from [11])**.** *A TEMPLOG clause C is said to be a* temporally ground instance *(TGI) iff:*

i If $C = \bigcirc^i A \leftarrow B$ is an initial clause then and $B^* \in TGI(B, 0)$ then $C^* = \bigcirc^i A \leftarrow B^*$ is a temporally ground instance of C.

ii *If* $C = \Box \bigcirc^i A \leftarrow B$ *is an initial clause and* $B^* \in TGI(B, 0)$ *then for every* $k \in \mathbb{N}$, $C^* = \bigcirc^{i+k} A \leftarrow B^*$ *is a TGI of* C.

iii *If* $C = \Box \left(\bigcirc^i A \leftarrow B\right)$ *is a permanent clause and* $B^* \in TGI(B, 0)$ *then for every* $k \in \mathbb{N}$, $C^* = \bigcirc^{i+k} A \leftarrow \bigcirc^k B^*$ *is a TGI of* C.

⊠

The following lemma states that Π^* can be seen as an extension of the $\| \ \|_0$ transformation applied to TEMPLOG programs, that is, $\|\Pi\|_0$ would result from applying the transformation to every rule in Π.

Lemma 7.1. *Let* Π *and* Π^* *be a TEMPLOG program and its corresponding TGI respectively. Given a HT*$^\infty$ *interpretation* $M^\infty = \langle H^\infty, T^\infty \rangle$*, with* $H^\infty \subseteq T^\infty \subseteq At^\infty$*, for any signature* At*, it holds that*

$$M^\infty \models \Pi^* \quad \textit{iff} \quad M^\infty \models \|\Pi\|_0$$

Proof. We proceed by checking cases i-iii of Definition 7.4.

$$\begin{aligned}
M^\infty \models i \Leftrightarrow\ & \forall B^* \in TGI(B,0),\ M^\infty \models \bigcirc^i A \leftarrow B^* \\
\Leftrightarrow\ & M^\infty \models \{\bigcirc^i A \leftarrow B^* \mid B^* \in TGI(B,0)\}^\wedge && \text{(Def. 2.25)} \\
\Leftrightarrow\ & M^\infty \models \bigcirc^i A \leftarrow \{B^* \mid B^* \in TGI(B,0)\}^\vee && \text{(Prop. 2.6)} \\
\Leftrightarrow\ & M^\infty \models \| \bigcirc^i A\|_0 \leftarrow \|B\|_0 && \text{(Prop. 7.1)} \\
\Leftrightarrow\ & M^\infty \models \| \bigcirc^i A \leftarrow B\|_0 && \text{(Def. 3.6)}
\end{aligned}$$

$$\begin{aligned}
M^\infty \models ii \Leftrightarrow\ & \forall k \geq 0\ \forall B^* \in TGI(B,0),\ M^\infty \models \bigcirc^{i+k} A \leftarrow B^* \\
\Leftrightarrow\ & \begin{cases} M^\infty \models \{\{\bigcirc^{i+k} A \leftarrow B^* \mid B^* \in TGI(B,0)\}^\wedge \\ \qquad\qquad\qquad\qquad \mid k \geq 0\}^\wedge \end{cases} && \text{(Def. 2.25)} \\
\Leftrightarrow\ & \begin{cases} M^\infty \models \{\bigcirc^{i+k} A \leftarrow \{B^* \mid B^* \in TGI(B,0)\}^\vee \\ \qquad\qquad\qquad\qquad \mid k \geq 0\}^\wedge \end{cases} && \text{(Prop. 2.6)} \\
\Leftrightarrow\ & M^\infty \models \{\bigcirc^{i+k} A \mid k \leq 0\}^\wedge \leftarrow \{B^* \mid B^* \in TGI(B,0)\}^\vee && \text{(Prop. 2.6)} \\
\Leftrightarrow\ & M^\infty \models \{\bigcirc^{i+k} A \mid k \leq 0\}^\wedge \leftarrow \|B\|_0 && \text{(Prop. 7.1)} \\
\Leftrightarrow\ & M^\infty \models \|\Box \bigcirc^i A\|_0 \leftarrow \|B\|_0 && \text{(Def. 3.6)} \\
\Leftrightarrow\ & M^\infty \models \|\Box \bigcirc^i A \leftarrow B\|_0 && \text{(Def. 3.6)}
\end{aligned}$$

$M^\infty \models iii \Leftrightarrow \forall k \geq 0\ \forall B^* \in TGI(B,0),\ M^\infty \models \bigcirc^{i+k} A \leftarrow \bigcirc^k B^*$

$$\Leftrightarrow \begin{cases} M^\infty \models \{\{\bigcirc^{i+k} A \leftarrow \bigcirc^k B^* \mid B^* \in TGI(B,0)\}^\wedge \\ \qquad\qquad\qquad\qquad\qquad\qquad \mid k \geq 0\}^\wedge \end{cases} \text{(Def. 2.25)}$$

$$\Leftrightarrow \begin{cases} M^\infty \models \{\bigcirc^{i+k} A \leftarrow \{\bigcirc^k B^* \mid B^* \in TGI(B,0)\}^\vee \\ \qquad\qquad\qquad\qquad\qquad\qquad \mid k \geq 0\}^\wedge \end{cases} \text{(Prop. 2.6)}$$

$$\begin{array}{ll} \Leftrightarrow M^\infty \models \{\bigcirc^{i+k} A \leftarrow \{\widehat{B} \mid \widehat{B} \in TGI(B,k)\}^\vee \mid k \geq 0\}^\wedge & \text{(Def. 7.2)} \\ \Leftrightarrow M^\infty \models \{\bigcirc^{i+k} A \leftarrow \|B\|_k \mid k \geq 0\}^\wedge & \text{(Prop. 7.1)} \\ \Leftrightarrow M^\infty \models \{\| \bigcirc^{i+k} A\|_0 \leftarrow \|B\|_k \mid k \geq 0\}^\wedge & \text{(Def. 3.6)}^1 \\ \Leftrightarrow M^\infty \models \{\| \bigcirc^{i} A\|_k \leftarrow \|B\|_k \mid k \geq 0\}^\wedge & \text{(Def. 3.6)} \\ \Leftrightarrow M^\infty \models \{\| \bigcirc^{i} A \leftarrow B\|_k \mid k \geq 0\}^\wedge & \text{(Def. 3.6)} \\ \Leftrightarrow M^\infty \models \|\Box\left(\bigcirc^i A \leftarrow B\right)\|_0 & \text{(Def. 3.6)} \end{array}$$

⊠

Definition 7.4 allows seeing any next-atom "$\bigcirc^i P$" as a new renamed classical atom, and considering an infinite signature of the shape $At = \{\bigcirc^i P \mid i \in \mathbb{N}\}$. Under this point of view, Π^* becomes a *definite* logic program (negation does not occur in next-atoms), and so, it is possible to define the least model of Π^*, as a classical program.

Theorem 7.1. *If* Π *is a TEMPLOG program and* L *is least model* Π^* *then, the temporal equilibrium model* $\mathbf{T}$*, defined as*

$$T_i = \{p \mid \bigcirc^i p \in L \textit{ with p an atom }\}$$

is the unique equilibrium model of Π.

Proof. Since Π^* is a definite positive program, its least model L is its unique stable model. Next, from [115] we derive that L is the unique infinitary stable model and, moreover, $\langle L, L\rangle$ is the unique infinitary equilibrium model (this result was also proved in [55]). Finally, the fact that $\langle L, L\rangle$ is the unique stable model of $\|\Pi\|_0$ follows from Lemma 7.1 and, because of Lemma 3.7, $\langle \mathbf{T}, \mathbf{T}\rangle$ is the unique stable model of Π ⊠

7.2 Relation to Temporal Answer Sets

In [51], Giordano, Martelli and Theseider Dupré defined an extension of answer sets semantics to deal with operators from dynamic linear temporal logic with the purpose of representing action scenarios.

7.2.1 Temporal Answer Sets

TAS syntax is based on two sets of atoms, one for *actions* and another for *fluents* denoted by Σ and Lit_S respectively. If $a \in \Sigma$ and $l \in Lit_S$ is any simple literal, then $\bigcirc l$ and $[a]l$ are called *temporal fluent literals* (Lit_T). Informally

[1] Note that, since A is an atom, then $\| \bigcirc^{i+k} A\|_0 \equiv \bigcirc^{i+k} A$

speaking $[a]l$ means that l has to be true after executing the action a whereas $\bigcirc l$ states that l will be true after the execution of any action. We further define the set $Lit \stackrel{\text{def}}{=} Lit_S \cup Lit_T \cup \{\bot\}$. An *extended* fluent literal is defined as either t or $not\ t$ with $t \in Lit$. TAS programs are defined as follows:

Definition 7.5 (domain description)**.** *A* domain description D *over* Σ *is a tuple* $D = (\Pi, \mathcal{C})$ *where* $\mathcal{C}$ *is a set of DLTL formulas called* constraints*, and* Π *is a set of rules of the forms:*

$$t_0 \leftarrow t_1, \ldots, t_m, not\ t_{m+1}, \ldots, not\ t_n \tag{7.1}$$

$$\Box\,(t_0 \leftarrow t_1, \ldots, t_m, not\ t_{m+1}, \ldots, not\ t_n) \tag{7.2}$$

with the following restrictions:

1. *If* $t_0 \in Lit_S$*, then all* $t_i \in Lit_S$ *for* $i = 1, \ldots, n$*.*
2. *If* $t_0 = \bigcirc l$*, all the temporal literals in the rule are of the form* $\bigcirc l'$
3. *If* $t_0 = [a]l$*, all the temporal literals in the rule are of the form* $[a]l'$

⊠

To give an example, let us consider following is a possible TAS encoding of the Yale Shooting scenario where we consider that the shot of a loaded gun can non-deterministically fail in killing the turkey:

$$\Box([shoot] \sim loaded \leftarrow loaded) \tag{7.3}$$

$$\Box([shoot](alive \vee \sim alive) \leftarrow alive \wedge loaded) \tag{7.4}$$

$$\Box[load]loaded \tag{7.5}$$

$$\Box(\bigcirc F \leftarrow F \wedge \neg \bigcirc \sim F) \tag{7.6}$$

$$\Box(\bigcirc \sim F \leftarrow \sim F \wedge \neg \bigcirc F) \tag{7.7}$$

where (7.6) and (7.7) represent the inertia for all fulent $F \in \{loaded, alive\}$.

Temporal connectives are interpreted in terms of words (including the empty word ϵ) build with symbols from Σ. Since the words considered can have either finite or infinite length, let Σ^* and Σ^ω be the sets of words with finite (resp. infinite) length that can be formed with symbols from Σ. Moreover given a word σ we call $pref(\sigma)$ to the set consisting of all prefixes of the word σ.

Definition 7.6 (partial temporal interpretation (from [51]))**.** *A* (partial) temporal interpretation *is a pair* (σ, S) *with* $\sigma \in \Sigma^\omega$ *and* S *a set of temporal expressions of the form* $[\tau]l$*, with* $\tau \in pref(\sigma)$ *and* $l \in Lit_S$*, not containing any pair* $[\tau]p$ *and* $[\tau] \sim p$ *for any* $p \in \mathcal{P}$*. Moreover, when* $[\tau]p \in S$ *iff* $[\tau] \sim p \notin S$*, then* (σ, S) *is a* total temporal interpretation*.*
The satisfiability of an extended fluent literal $t \in Lit$ *with respect to an interpretation* (σ, S) *and a prefix* $\tau \in pref(\sigma)$ *is defined as follows, depending on the case:*

- $(\sigma, S), \tau \models \top$ *and* $(\sigma, S), \tau \not\models \bot$

- $(\sigma, S), \tau \models l$ *iff* $[\tau]l \in S$*, for a simple literal* l
- $(\sigma, S), \tau \models [a]l$ *iff* $[\tau; a]l \in S$ *or* $\tau \notin pref(\sigma)$
- $(\sigma, S), \tau \models \bigcirc l$ *iff* $[\tau; b]l \in S$ *for some* $\tau; b \in pref(\Sigma)$

for any $a \in \Sigma$ *and* $l \in Lit_S$*. The satisfaction of a rule with respect to* (σ, S) *is as follows:*

- $(\sigma, S), a_1; \cdots, a_n \models$ (7.1) *when: if* $(\sigma, S), \epsilon \models t_i$ *for all* $i = 1, \ldots, m$ *and* $(\sigma, S), \epsilon \not\models t_i$ *for all* $i = m+1, \ldots, n$ *then* $(\sigma, S), \epsilon \models t_0$*.*
- $(\sigma, S), a_1, \cdots, a_n \models$ (7.2) *when: if for any* $\tau \in pref(\sigma)$ *such that* $(\sigma, S), \tau \models t_i$ *for all* $i = 1, \ldots, m$ *and* $(\sigma, S), \tau \not\models t_i$ *for all* $i = m+1, \ldots, n$ *then* $(\sigma, S), \tau \models t_0$*.*

⊠

The definition of Temporal Answer Sets is given in terms of an extension of Gelfond & Lifschitz's program reduct [48] to the temporal case

Definition 7.7 (program Reduct from [51])**.** *Given a partial temporal interpretation* (σ, S) *and a program* Π*, the program reduct, denoted as* $\Pi^{(\sigma,S)}$ *consists of the set of all rules*

$$\bigcup_{\tau \in pref(\sigma)} \{[\tau]\,(t_0 \leftarrow t_1, \ldots, t_m)\}$$

such that $\Box\,(t_0 \leftarrow t_1, \ldots, t_m, not\ t_{m+1}, \ldots, not\ t_n) \in \Pi$ *and* $(\sigma, S), \tau \models t_i$*, for all* $m \in (m, n]$*.* ⊠

Definition 7.8 (temporal answer set (from [51]))**.** *A partial temporal extension* (σ, S) *is a* temporal answer set *of* Π *if* (σ, S) *is minimal among the* S' *such that* (σ, S') *is a partial interpretation satisfying the rules of the reduct* $\Pi^{(\sigma,S)}$*.* ⊠

As an intuitive idea of whether a TAS program Π can be encoded in TEL or not, our Yale shooting representation has a simple encoding in TEL. It suffices with introducing new propositions for all the actions in Σ and adding them in the positive bodies where they occur.

$$\Box\,(shoot \wedge loaded \rightarrow \bigcirc \sim loaded) \tag{7.8}$$
$$\Box\,(shoot \wedge loaded \wedge alive \rightarrow \bigcirc alive \vee \bigcirc \sim alive) \tag{7.9}$$
$$\Box\,(load \rightarrow \bigcirc loaded) \tag{7.10}$$
$$\Box\,(F \wedge \neg \bigcirc \sim F \rightarrow \bigcirc F) \tag{7.11}$$
$$\Box\,(\sim F \wedge \neg \bigcirc F \rightarrow \bigcirc \sim F) \tag{7.12}$$
$$\Box\,(load \vee shoot) \tag{7.13}$$

Rule (7.13) is added to represent that one (and, in fact, only one) action can be executed at a time. Temporal equilibrium models of such representation, which correspond, with the temporal answer sets, are shown in Figure 7.1.

The relation between TAS and TEL has been deeply studied by Aguado, Pérez and Vidal in [4]. Such study was made by defining a new framework called *Dynamic Temporal Equilibrium Logic* (DTEL), the combination of Equilibrium Logic and DLTL, and proving both TEL and TAS can be embedded in DTEL.

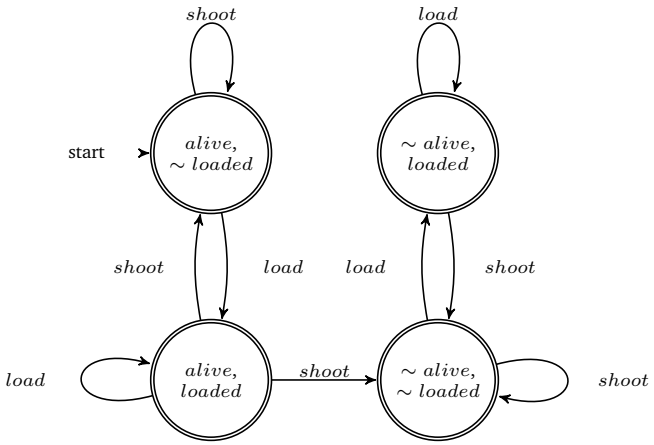


Figure 7.1: Temporal answer sets of program consisting of rules (7.3)-(7.7)

7.2.2 Dynamic Temporal Equilibrium Logic

In this section we recall the definition of made by Aguado, Pérez and Vidal, *Dynamic Linear-Time Temporal Equilibrium Logic* (DTEL for short). As happens with Equilibrium Logic and with TEL, DTEL is a non-monotonic formalism whose definition consists of two parts: a monotonic basis and a models selection criterion. The monotonic basis is a temporal extension of the intermediate logic of *Here-and-There* [57] (HT). We call this monotonic logic DLTL_{HT}. As a running example, we will use the well-known Yale Shooting scenario from [54] where, in order to kill a turkey, we must shoot a gun that must be previously loaded.

DLTL_{HT} formulas are built from a non-empty finite set of *actions*, called *alphabet* and denoted by Σ, and a set of *fluents*, denoted by $\mathcal{P}$, such that $\Sigma \cap \mathcal{P} = \emptyset$. We denote as Σ^* and Σ^ω to respectively stand for the finite and the non-finite words that can be formed with Σ. We also define $\Sigma^\infty \stackrel{\text{def}}{=} \Sigma^* \cup \Sigma^\omega$. For any $\sigma \in \Sigma^\omega$, we denote by $pref(\sigma)$ the set of its finite prefixes (including the empty word ϵ) and we say that two prefixes, τ_1 and τ_2, from $pref(\sigma)$ satisfy $\tau_1 < \tau_2$ iff

$$\exists \tau' \in \Sigma^* \text{ s. t. } \tau_1 \tau' = \tau_2.$$

In the same way, we say that $\tau_1 \leq \tau_2$ if $\tau_1 \leq \tau_2$ or $\tau_1 = \tau_2$.

The set of *programs* (regular expressions) generated by Σ is denoted by $Prg(\Sigma)$ and its syntax is given by the grammar:

$$\pi ::= a \,|\, \pi_0 + \pi_1 \,|\, \pi_0; \pi_1 \,|\, \pi^* \tag{7.14}$$

with $a \in \Sigma$ and $\pi, \pi_0, \pi_1 \in Prg(\Sigma)$. By abuse of notation, we will sometimes identify a finite prefix $\tau = \sigma_1 \ldots \sigma_n$ as the program $\sigma_1; \ldots; \sigma_n$. For example, in the case of the Yale Shooting scenario for the set of actions $\Sigma = \{load, shoot, wait\}$ we could write a program like $\pi = (load; shoot)^*$ representing repetitions of the sequence $load; shoot$.

The mapping $||\cdot|| : Prg(\Sigma) \to 2^{\Sigma^*}$ associates to each program a set of finite words (regular set) as follows:

$$\begin{array}{lcl} ||a|| & \stackrel{\text{def}}{=} & \{a\} \\ ||\pi_0 + \pi_1|| & \stackrel{\text{def}}{=} & ||\pi_0|| \cup ||\pi_1|| \\ ||\pi_0; \pi_1|| & \stackrel{\text{def}}{=} & \{\tau_0\tau_1 \,|\, \tau_0 \in ||\pi_0|| \text{ and } \tau_1 \in ||\pi_1||\} \\ ||\pi^*|| & \stackrel{\text{def}}{=} & \bigcup_{i\in\omega} ||\pi^i|| \end{array}$$

where

$$\begin{array}{lcl} ||\pi^0|| & \stackrel{\text{def}}{=} & \{\epsilon\} \\ ||\pi^{i+1}|| & \stackrel{\text{def}}{=} & \{\tau_0\tau_1 \,|\, \tau_0 \in ||\pi|| \text{ and } \tau_1 \in ||\pi^i||\} \text{ for every } i \in \omega. \end{array}$$

Let $\mathcal{P} = \{p_1, p_2, \ldots\}$ be a countable set of atomic propositions. We denote the set of *simple literals* as $Lit_S \stackrel{\text{def}}{=} \{p, {\sim}p \,;\, p \in \mathcal{P}\}$. The syntax of DLTL$_{\text{HT}}$ coincides with DLTL plus the addition of the strong negation operator '$\sim$.' A *well-formed formula* F is defined as follows:

$$F ::= \bot \,|\, p \,|\, {\sim}F \,|\, F_1 \vee F_2 \,|\, F_1 \wedge F_2 \,|\, F_1 \to F_2 \,|\, F_1\, \mathcal{U}^\pi\, F_2 \,|\, F_1\, \mathcal{R}^\pi\, F_2$$

where p is an atom and F, F_1, F_2 are well-formed formulas. The expression $\neg F$ stands for $F \to \bot$, constant $\top$ corresponds to $\neg\bot$ whereas $F_1 \leftrightarrow F_2$ is an abbreviation for $(F_1 \to F_2) \wedge (F_2 \to F_1)$ as usual.

Proposition 7.2 (from [4])**.** *A DLTL$_{HT}$ formula φ can be always rewritten into an equivalent one in* strong negation normal form *(SNNF), that is, guaranteeing that the operator '$\sim$' only affects to atoms in $\mathcal{P}$.* ⊠

Given an infinite word $\sigma \in \Sigma^\omega$, we define a *valuation function* V as a mapping $V : pref(\sigma) \to 2^{Lit_S}$ assigning a set of literals to each finite prefix of σ. A valuation function V is *consistent* if, for any $\tau \in pref(\sigma)$, $V(\tau)$ does not contain a pair of opposite literals of the form p and ${\sim}p$ simultaneously. Given two valuation functions V_1, V_2 (wrt σ), we write $V_1 \leq V_2$ when $V_1(\tau) \subseteq V_2(\tau)$ for every $\tau \in pref(\sigma)$. As usual, if $V_1 \leq V_2$ but $V_1 \neq V_2$, we just write $V_1 < V_2$.

Definition 7.9 (temporal interpretation)**.** *A* (temporal) interpretation *of DLTL$_{HT}$ is a tuple $M = (\sigma, V_h, V_t)$ where $\sigma \in \Sigma^\omega$ and V_h, V_t are two valuation functions for σ such that V_t is consistent and $V_h \leq V_t$. We say that the interpretation M is* total *when $V_h = V_t$.* ⊠

Given a formula α, a prefix $\tau \in pref(\sigma)$ and a temporal interpretation $M = (\sigma, V_h, V_t)$, we define the satisfaction relation $M, \tau \models \alpha$ inductively as follows:

- $M, \tau \models p$ iff $p \in V_h(\tau)$
- $M, \tau \models {\sim}p$ iff ${\sim}p \in V_h(\tau)$
- $M, \tau \models \alpha \vee \beta$ iff $M, \tau \models \alpha$ or $M, \tau \models \beta$
- $M, \tau \models \alpha \wedge \beta$ iff $M, \tau \models \alpha$ and $M, \tau \models \beta$

- $M,\tau \models \alpha \rightarrow \beta$ iff for every $w \in \{h,t\}$, $(\sigma, V_w, V_t),\tau \not\models \alpha$ or $(\sigma, V_w, V_t),\tau \models \beta$
- $M,\tau \models \alpha\,\mathcal{U}^{\pi}\,\beta$ iff there exists $\tau' \in ||\pi||$ such that $\tau\tau' \in pref(\sigma)$ and $M,\tau\tau' \models \beta$, and for every τ'' such that $\epsilon \leq \tau'' < \tau'$, we have $M,\tau\tau'' \models \alpha$.
- $M,\tau \models \alpha\,\mathcal{R}^{\pi}\,\beta$ iff for every $\tau' \in ||\pi||$ such that $\tau\tau' \in pref(\sigma)$, it is the case that $M,\tau\tau' \models \beta$ or there exists τ'' such that $\epsilon \leq \tau'' < \tau'$ and $M,\tau\tau'' \models \alpha$.

It has also been proven that persistence also holds in this formalism:

Lemma 7.2 (persistency from [4]). *For any formula α, any interpretation $M = (\sigma, V_h, V_t)$ and any $\tau \in pref(\sigma)$, the following two conditions hold:*

1. *If $(\sigma, V_h, V_t), \tau \models \alpha$, then $(\sigma, V_t, V_t), \tau \models \alpha$*
2. *$(\sigma, V_h, V_t), \tau \models \neg\alpha$ iff $(\sigma, V_t, V_t), \tau \not\models \alpha$*

⊠

Other usual temporal modalities can be defined as derived operators:

(i) $\langle\pi\rangle\,\alpha \stackrel{\text{def}}{=} \top\,\mathcal{U}^{\pi}\alpha$

(ii) $[\pi]\,\alpha \stackrel{\text{def}}{=} \bot\,\mathcal{R}^{\pi}\alpha$

(iii) $\bigcirc\alpha \stackrel{\text{def}}{=} \bigvee_{a\in\Sigma}\langle a\rangle\,\alpha$

(iv) $\alpha\,\mathcal{U}\,\beta \stackrel{\text{def}}{=} \alpha\,\mathcal{U}^{\Sigma^*}\,\beta$

(v) $\alpha\,\mathcal{R}\,\beta \stackrel{\text{def}}{=} \alpha\,\mathcal{R}^{\Sigma^*}\,\beta$

(vi) $\Box\alpha \stackrel{\text{def}}{=} [\Sigma^*]\,\alpha \;\; (\equiv \bot\,\mathcal{R}\,\alpha)$

(vii) $\Diamond\alpha \stackrel{\text{def}}{=} \langle\Sigma^*\rangle\,\alpha \;\; (\equiv \top\,\mathcal{U}\,\alpha)$

Definition 7.10 (DTEL model). *A total temporal interpretation $M = (\sigma, V_t, V_t)$ is said to be a* temporal equilibrium model *(DTEL-model for short) of a formula α in DLTL$_{HT}$ if $M \models \alpha$ and there is no $V_h < V_t$ such that $(\sigma, V_h, V_t) \models \alpha$.*

When a formula φ is written in LTL syntax, Theorem 7.2 shows that there exist an one-to-one correspondence between both THT and DLTL$_{\text{HT}}$ models of φ, which still holds after the minimisation.

Theorem 7.2 (from [4]). *Let α be a formula in LTL syntax. Then the THT models of α are in a one-to-one correspondence to (an equivalence class of) the DLTL$_{HT}$ models of α.* ⊠

For embedding TAS in DLTL$_{\text{HT}}$, we have to take into account that the formulas of $\mathcal{C}$ are also formulas of DLTL$_{\text{HT}}$ and translate the rules of Π into formulas in DLTL$_{\text{HT}}$. For any rule r, we define the formula $\widetilde{r}$ as:

$$\widetilde{r} \stackrel{\text{def}}{=} t_1 \wedge \ldots \wedge t_m \wedge \neg t_{m+1} \wedge \ldots \wedge \neg t_n \rightarrow t_0 \tag{7.15}$$

$$\widetilde{r} \stackrel{\text{def}}{=} \Box\,(t_1 \wedge \ldots \wedge t_m \wedge \neg t_{m+1} \wedge \ldots \wedge \neg t_n \rightarrow t_0) \tag{7.16}$$

for r of the forms (7.1) and (7.2), respectively. Note that for positive programs, $m = n$ and the empty conjunction of negated t_i amounts to $\top$.

On the other hand, we also have to define a correspondence between a partial interpretation (σ, S) and a DLTL$_{\text{HT}}$ total interpretation. Such a correspondence is defined by taking V_S and S as follows:

$$\begin{aligned} V_S(\tau) &\stackrel{\text{def}}{=} \{l \in Lit_S \mid [\tau]\, l \in S\} \\ S &\stackrel{\text{def}}{=} \bigcup_{\tau \in pref(\sigma)} \{[\tau]\, V_S(\tau)\} \end{aligned}$$

Theorem 7.3 (from [4]). *Take $D = (\Pi, \mathcal{C})$ a domain description and (σ, S) a temporal interpretation. The following assertions are equivalent:*

1. *(σ, S) is an extension of D*
2. *(σ, V_S, V_S) is a temporal equilibrium model of $\widetilde{\Pi} \cup \{\neg\neg\alpha \mid \alpha \in \mathcal{C}\}$*

⊠

7.2.3 A normal form for DTEL

We provide next a new result showing how can we remove DTEL modalities by allowing the use of auxiliary atoms[2]. Both formalisms, DTEL and TEL, have different expressive powers, while the former allows describing problems that are representable by a ω-regular expression, the latter covers a less expressive class that corresponds to the star-free ω-languages. For instance, the well known example of even-state property is represented by the expression $[(\Sigma;\Sigma)^*]p$ it is not TEL representable.

In order to increase such expressiveness, DTEL introduces two new modalities, $\mathcal{U}^\pi$ and $\mathcal{R}^\pi$, that allow forcing that DTEL models satisfy π. However, if we allow the use of auxiliary atoms we can reduce an arbitrary DLTL$_{\text{HT}}$ theory to a set of implications involving only temporal operators $\Box$, $\bigcirc$, $[\pi^*]$ and $\langle\pi^*\rangle$. We assume that DLTL$_{\text{HT}}$ input formulas are given in the normal form defined next.

Definition 7.11 (DLTL$_{\text{HT}}$ normal form). *We say that a DLTL$_{HT}$ formula φ, is in* normal form *if every subformula of φ involving operators $\mathcal{U}$ and $\mathcal{R}$ has the following form*

- $\varphi \mathcal{U}^a\ \psi$, *with* $a \in \Sigma$
- $\varphi \mathcal{R}^a\ \psi$, *with* $a \in \Sigma$
- $\varphi \mathcal{U}^{\pi^*}\ \psi$
- $\varphi \mathcal{R}^{\pi^*}\ \psi$

An arbitrary DLTL$_{\text{HT}}$ formula φ can be translated into this normal form by applying iteratively equivalences (1)-(4) from Definition 7.12 to all subformulas of φ.

[2] An auxiliary atom is a fresh new atom that is disregarded once the final models are obtained. In this sense it behaves exactly as an existential quantifier (this was suggested by Stéphane Demri in a personal communication)

Definition 7.12. *The following equivalences are valid in DLTL$_{HT}$*

1 $\varphi\mathcal{U}^{\pi_1;\pi_2}\ \psi \equiv \varphi\mathcal{U}^{\pi_1}\ (\varphi\mathcal{U}^{\pi_2}\ \psi)$

2 $\varphi\mathcal{R}^{\pi_1;\pi_2}\ \psi \equiv \varphi\mathcal{R}^{\pi_1}\ (\varphi\mathcal{R}^{\pi_2}\ \psi)$

3 $\varphi\mathcal{U}^{\pi_1+\pi_2}\ \psi \equiv (\varphi\mathcal{U}^{\pi_1}\ \psi) \vee (\varphi\mathcal{U}^{\pi_2}\ \psi)$

4 $\varphi\mathcal{R}^{\pi_1+\pi_2}\ \psi \equiv (\varphi\mathcal{R}^{\pi_1}\ \psi) \wedge (\varphi\mathcal{R}^{\pi_2}\ \psi)$

Translating star-free expressions

We first begin with the cases where regular expressions associated with operators $\mathcal{U}^{\pi}$ and $\mathcal{R}^{\pi}$ are star-free, that is when π^* is removed from the grammar (7.14). In this case, normal form of Definition 7.11 would contain only temporal operators of the form $\mathcal{U}^a$ and $\mathcal{R}^a$, with a an action. In this case, the following definition translates a DLTL$_{\text{HT}}$ expression, given in normal form, into THT without introducing auxiliary atoms.

Definition 7.13. *Let φ a DLTL$_{HT}$ formula expressed in normal form. Its translation into THT is recursively defined next:*

- $[\bot]_{THT} \stackrel{\text{def}}{=} \bot$
- $[p]_{THT} \stackrel{\text{def}}{=} p$*, with p an atom.*
- $[\bigcirc\varphi]_{THT} \stackrel{\text{def}}{=} \bigcirc[\varphi]_{THT}$
- $[\varphi \wedge \psi]_{THT} \stackrel{\text{def}}{=} [\varphi]_{THT} \wedge [\psi]_{THT}$
- $[\varphi \vee \psi]_{THT} \stackrel{\text{def}}{=} [\varphi]_{THT} \vee [\psi]_{THT}$
- $[\varphi \rightarrow \psi]_{THT} \stackrel{\text{def}}{=} [\varphi]_{THT} \rightarrow [\psi]_{THT}$
- $[\varphi\mathcal{U}^a\psi]_{THT} \stackrel{\text{def}}{=} [\varphi]_{THT} \wedge a \wedge \bigcirc[\psi]_{THT}$*, with a and action*
- $[\varphi\mathcal{R}^a\psi]_{THT} \stackrel{\text{def}}{=} a \rightarrow [\varphi]_{THT} \vee \bigcirc[\psi]_{THT}$*, with a an atom*

⊠

We can stablish the following one-to-one correspondence between DLTL$_{\text{HT}}$ and THT interpretations

Definition 7.14. *Let Σ and $\mathcal{P}$ a set of actions and propositional variables respectively. Given a DLTL$_{HT}$ interpretation $M = (a_0 \cdots a_i, V_h, V_t)$, we define the corresponding THT interpretation $M' = \langle H, T\rangle$ as:*

$$\begin{aligned} H_i &= \{a_{i+1}\} \cup V_h(a_0; \cdots; a_i) \\ T_i &= \{a_{i+1}\} \cup V_t(a_0; \cdots; a_i) \end{aligned}$$

Conversely, M can be obtained from M' by defining

$$\begin{aligned} a_i &= H_i \cap \Sigma \\ V_h(a_0 \cdots a_i) &= H_i \cap \mathcal{P} \\ V_t(a_0 \cdots a_i) &= T_i \cap \mathcal{P} \end{aligned}$$

⊠

Lemma 7.3. *Let φ be a DLTL$_{HT}$ expressed in normal form such that it has no subformulas of the type $\mathcal{U}^{\pi^*}$ and $\mathcal{R}^{\pi^*}$. For any model $M = (a_0 \cdots a_i, V_h, V_t)$ of φ and its corresponding THT model $M' = \langle H, T \rangle$, it holds that*

$$M,\ a_0 \cdots a_i \models \varphi \textit{ iff } M', i \models [\varphi]_{THT}.$$

Proof. We proceed by structural induction:

$$\begin{aligned} M, a_0 \cdots a_i \models p \in \mathcal{P} \quad &\Leftrightarrow \quad p \in V_h(a_0 \cdots a_i) \\ &\Leftrightarrow \quad p \in H_i && \text{(Def. 7.14)} \\ &\Leftrightarrow \quad [p]_{THT} \in H_i && \text{(Def. 7.13)} \\ &\Leftrightarrow \quad M', i \models [p]_{THT} \end{aligned}$$

$$\begin{aligned} M, a_0 \cdots a_i \models \varphi \wedge \psi \quad &\Leftrightarrow \quad M, a_0 \cdots a_i \models \varphi \text{ and } M, a_0 \cdots a_i \models \psi \\ &\Leftrightarrow \quad M', i \models [\varphi]_{THT} \text{ and } M', i \models [\psi]_{THT} && \text{(ind.)} \\ &\Leftrightarrow \quad M', i \models [\varphi \wedge \psi]_{THT} \end{aligned}$$

$$\begin{aligned} M, a_0 \cdots a_i \models \varphi \vee \psi \quad &\Leftrightarrow \quad M, a_0 \cdots a_i \models \varphi \text{ or } M, a_0 \cdots a_i \models \psi \\ &\Leftrightarrow \quad M', i \models [\varphi]_{THT} \text{ or } M', i \models [\psi]_{THT} && \text{(ind.)} \\ &\Leftrightarrow \quad M', i \models [\varphi \vee \psi]_{THT} \end{aligned}$$

For the case of the implication, let us denote as $\mathcal{M} = (a_0 \cdots a_i, V_t, V_t)$ and $\mathcal{M}' = \langle T, T \rangle$ the total models corresponding to M and M' respectively. Note that, since $\mathcal{M}$ also corresponds to $\mathcal{M}'$, we can apply induction on subformulas to prove this case.

$$\begin{aligned} M, a_0 \cdots a_i \models \varphi \rightarrow \psi \quad &\Leftrightarrow \quad \begin{cases} M, a_0 \cdots a_i \not\models \varphi \text{ or } M, a_0 \cdots a_i \models \psi \\ \text{and} \\ \mathcal{M}, a_0 \cdots a_i \not\models \varphi \text{ or } a_0 \cdots a_i \models \psi \end{cases} \\ &\Leftrightarrow \quad \begin{cases} M', i \not\models [\varphi]_{THT} \text{ or } M', i \models [\psi]_{THT} \\ \text{and} \\ \mathcal{M}', i \not\models [\varphi]_{THT} \text{ or } \mathcal{M}', i \models [\psi]_{THT} \end{cases} && \text{(Ind.)} \\ &\Leftrightarrow \quad M', i \models [\varphi \rightarrow \psi]_{THT} \end{aligned}$$

$$
\begin{array}{lcll}
M, a_0 \cdots a_i \models \bigcirc\varphi & \Leftrightarrow & \exists b \in \Sigma \text{ s.t. } M, a_0 \cdots a_i b \models \varphi & \\
& \Leftrightarrow & M', i+1 \models [\varphi]_{THT} & \text{(Ind.)} \\
& \Leftrightarrow & M', i \models \bigcirc[\varphi]_{THT} & \\
& \Leftrightarrow & M', i \models [\bigcirc\varphi]_{THT} &
\end{array}
$$

$$
\begin{array}{lcll}
M, a_0 \cdots a_i \models \varphi \mathcal{U}^b \psi & \Leftrightarrow & \begin{cases} M, a_0 \cdots a_i b \models \psi \\ \text{and } M, a_0 \cdots a_i \models \varphi \end{cases} & \\
& \Leftrightarrow & \begin{cases} M', i+1 \models [\psi]_{THT} \\ \text{and } M', i \models [\varphi]_{THT} \end{cases} & \text{(Ind.)} \\
& \Leftrightarrow & \begin{cases} M', i+1 \models [\psi]_{THT} \\ \text{and } M', i \models [\varphi]_{THT} \\ \text{and } M', i \models b \end{cases} & \text{(Def. 7.14)} \\
& \Leftrightarrow & \begin{cases} M', i \models \bigcirc[\psi]_{THT} \\ \text{and } M', i \models [\varphi]_{THT} \\ \text{and } M', i \models b \end{cases} & \\
& \Leftrightarrow & M', i \models [\varphi \mathcal{U}^b \psi]_{THT} &
\end{array}
$$

$$
\begin{array}{lcll}
M, a_0 \cdots a_i \models \varphi \mathcal{R}^b \psi & \Leftrightarrow & \begin{cases} \text{if } a_0 \cdots a_i b \in pref(\sigma) \text{ then} \\ M, a_0 \cdots a_i b \models \psi \\ \text{or } M, a_0 \cdots a_i \models \varphi \end{cases} & \\
& \Leftrightarrow & \begin{cases} \text{if } a_0 \cdots a_i b \in pref(\sigma) \text{ then} \\ M', i+1 \models [\psi]_{THT} \\ \text{or } M, i \models [\varphi]_{THT} \end{cases} & \text{(Ind.)} \\
& \Leftrightarrow & \begin{cases} \text{if } b \in H_i \text{ then} \\ M', i+1 \models [\psi]_{THT} \\ \text{or } M', i \models [\varphi]_{THT} \end{cases} & \text{(Def. 7.14)} \\
& \Leftrightarrow & \begin{cases} \text{if } M', i \models b \text{ then} \\ M', i+1 \models [\psi]_{THT} \\ \text{or } M', i \models [\varphi]_{THT} \end{cases} & \text{(Def. 3.1)} \\
& \Leftrightarrow & M', i \models b \rightarrow \bigcirc[\psi]_{THT} \vee [\varphi]_{THT} & \\
& \Leftrightarrow & M', i \models [\varphi R^b \psi]_{THT} &
\end{array}
$$

⊠

Going back to our running example, imagine that we want to select models where the shooter loads the gun once or twice in a row before shooting

the well-aimed shot that kills the turkey. This behaviour corresponds to the DLTL_{HT} formula

$$\gamma = \Diamond\langle(load + (load; load))\,; shoot\rangle \sim alive$$

which could be translated into THT as follows:

$$\Diamond\langle(load+(load; load;))\,; shoot\rangle \sim alive \Leftrightarrow \Diamond\langle load+(load; load)\rangle\langle shoot\rangle \sim alive$$

$$\Leftrightarrow \Diamond\left(\langle load;\ shoot\rangle \sim alive\right) \vee \Diamond\left(\langle load;\ load;\ shoot\rangle \sim alive\right)$$

$$\Leftrightarrow \begin{array}{l}\gamma' = \Diamond\left(load \wedge \bigcirc shoot \wedge \bigcirc\bigcirc \sim alive\right)\\ \vee\Diamond\left(load \wedge \bigcirc load \wedge \bigcirc\bigcirc shoot \wedge \bigcirc\bigcirc\bigcirc \sim alive\right).\end{array}$$

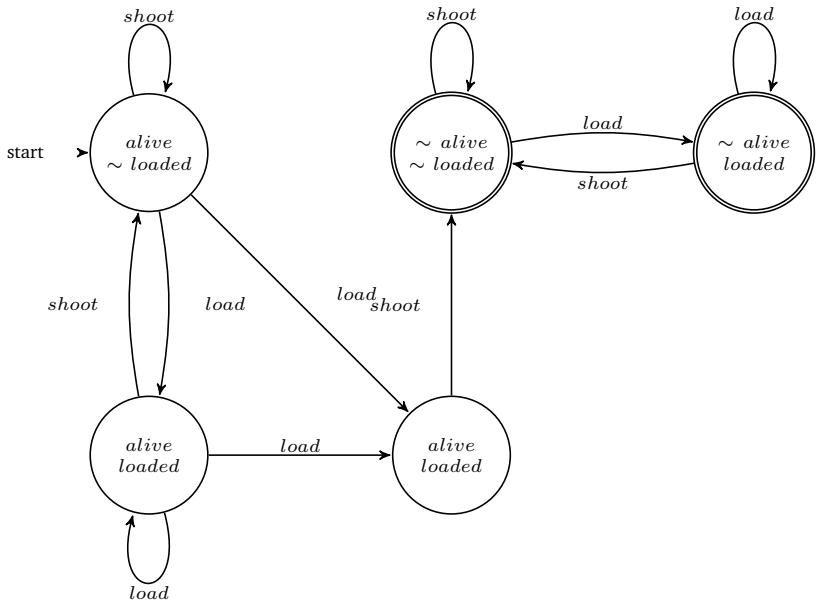


Figure 7.2: DTEL models of theory (7.8)-(7.13) plus γ'

Removing DTEL modalities

When operators $\mathcal{U}^{\pi^*}$ or $\mathcal{R}^{\pi^*}$ are used in the normal form, the resulting formulas may not be directly translated into TEL. Some example like $\alpha\mathcal{U}^{a^*}\beta$ is equivalent to the THT expression $(\alpha \wedge a)\mathcal{U}\beta$ but more general regular expressions, like $(a; b)^*$ are not representable. If we allow the use of auxiliary atoms we can reduce the set of DLTL_{HT} modalities to the cases $\langle\pi^*\rangle$ and $[\pi^*]$.

Definition 7.15. *Let $\varphi\mathcal{U}^{\pi^*}\psi$ and $\varphi\mathcal{R}^{\pi^*}\psi$ be two $DLTL_{HT}$ formulas built on a set of actions Σ and a set of atoms $\mathcal{P}$. We call their* definitions *respectively, denoted*

by $df(\varphi\mathcal{U}^{\pi^*}\psi)$ and $df(\varphi\mathcal{R}^{\pi^*}\psi)$, to the following axioms

$$\begin{aligned} df(\varphi\mathcal{U}^{\pi^*}\psi) &\stackrel{\text{def}}{=} \Box(\mathbf{L}_{\varphi\mathcal{U}^{\pi^*}\psi} \leftrightarrow \psi \vee (\varphi\mathcal{U}^{\pi}\mathbf{L}_{\varphi\mathcal{U}^{\pi^*}\psi})) \\ &\quad \wedge\Box(\mathbf{L}_{\varphi\mathcal{U}^{\pi^*}\psi} \rightarrow \langle\pi^*\rangle\psi) \qquad (7.17) \\ df(\varphi\mathcal{R}^{\pi^*}\psi) &\stackrel{\text{def}}{=} \Box(\mathbf{L}_{\varphi\mathcal{R}^{\pi^*}\psi} \leftrightarrow \psi \wedge (\varphi\mathcal{R}^{\pi}\mathbf{L}_{\varphi\mathcal{R}^{\pi^*}\psi}) \\ &\quad \wedge\Box([\pi^*]\psi \rightarrow \mathbf{L}_{\varphi\mathcal{R}^{\pi^*}\psi}) \qquad (7.18) \end{aligned}$$

where both $\mathbf{L}_{\varphi\mathcal{U}^{\pi^*}\psi}$ and $\mathbf{L}_{\varphi\mathcal{R}^{\pi^*}\psi}$ are new fresh variables that represent auxiliary atoms. ⊠

Lemmas 7.4 and 7.5, which are presented below, guarantee that replacing a formula of the form $\alpha = \varphi\mathcal{U}^{\pi^*}\psi$ or $\alpha = \varphi\mathcal{R}^{\pi^*}\psi$ by a fresh atom $\mathbf{L}_\alpha$ plus adding corresponding definitions $df(\varphi)$ to the context, we can reduce the corresponding DLTL$_{\text{HT}}$ modalities to the case of $[\pi^*]$ and $\langle\pi^*\rangle$. Doing this process iteratively we reduce all modalities.

Lemma 7.4. *Let Σ be a set of actions, $\mathcal{L}_V$ a set of atoms and φ and ψ two DLTL$_{HT}$ formulas over $\mathcal{L}_V$. If $\alpha = \varphi\mathcal{U}^{\pi^*}\psi$ and $\mathcal{L}_U = \mathcal{L}_V \cup \{\mathbf{L}_\alpha\}$ (with $\mathbf{L}_\alpha$ a fresh atom), given a DLTL$_{HT}$ interpretation $M = (\sigma, V_h, V_t)$ on $\mathcal{L}_U$, it holds that*

$$M, \tau \models \mathbf{L}_\alpha \textit{ iff } M, \tau \models \alpha$$

for any $\tau \in pref(\sigma)$.

Proof. From left to right, given $M, \tau \models \mathbf{L}_\alpha$ then, due to expression (7.15) we get $M, \tau \models \langle\pi^*\rangle\psi$ and, thus, there exist $i \geq 0$ satisfying $M, \tau \models \langle\pi^i\rangle\psi$. Take the shortest i satisfying $M, \tau \models \langle\pi^i\rangle\psi$, so that we further have $M, \tau \not\models \langle\pi^j\rangle\psi$, for any $j < i$. We will inductively prove that $M, \tau \models \varphi\mathcal{U}^{\pi^k}\mathbf{L}_\alpha$ with $k = 0, \cdots i-1$, which together with $M, \tau \models \langle\pi^i\rangle\psi$ implies $M, \tau \models \alpha$.

For $k = 0$, we know $M, \tau \models \mathbf{L}_\psi$ thus $M, \tau \models \varphi\mathcal{U}^{\pi^0}\mathbf{L}_\alpha$. Assume it proved for a program π^k, with $0 \leq k < i-1$ and let us prove it for $k+1$. By induction, $M, \tau \models \varphi\mathcal{U}^{\pi^k}\mathbf{L}_\alpha$ and so $M, \tau \models \langle\pi^k\rangle\mathbf{L}_\alpha$. Since $M, \tau \models df(\alpha)$ and $M, \tau \not\models \langle\pi^k\rangle\psi$ we deduce that $M, \tau \models \varphi\mathcal{U}^{\pi}\mathbf{L}_\alpha$ and therefore $M, \tau \models \varphi\mathcal{U}^{\pi^k}(\varphi\mathcal{U}^{\pi}\mathbf{L}_\alpha)$, which is equivalent to $M, \tau \models \varphi\mathcal{U}^{\pi^{k+1}}\mathbf{L}_\alpha$.

From right to left, suppose $M, \tau \models \varphi\mathcal{U}^{\pi^*}\psi$. This means that there exists $i \geq 0$ such that $M, \tau \models \varphi\mathcal{U}^{\pi^i}\psi$. We will inductively show that for any $k = i, \cdots, 0$ it holds that $M, \tau \models \langle\pi^k\rangle\mathbf{L}_\alpha$, which includes the case $k = 0$, which is the one we really want to prove. For $k = i$, we know that $M, \tau \models \langle\pi^i\rangle\mathbf{L}_\alpha$ and, by expression (7.17), we derive $M, \tau \models \langle\pi^i\rangle\mathbf{L}_\alpha$.

Assume it proved for $k+1$, with $0 \leq k < j$ and let us prove it for k. Since $M, \tau \models \varphi\mathcal{U}^{\pi^i}\psi$ can be rewritten as

$$M, \tau \models \varphi\mathcal{U}^{\pi^{k+1}}\left(\varphi\mathcal{U}^{\pi^{i-k-1}}\psi\right),$$

and $M, \tau \models \langle\pi^{k+1}\rangle\mathbf{L}_\alpha$, we can derive $M, \tau \models \varphi\mathcal{U}^{\pi^{k+1}}\mathbf{L}_\alpha$. Again, this expression is equivalent to say $M, \tau \models \varphi\mathcal{U}^{\pi^k}(\varphi\mathcal{U}^{\pi}\mathbf{L}_\alpha)$, which implies $M, \tau \models \langle\pi^k\rangle(\varphi\mathcal{U}^{\pi}\mathbf{L}_\alpha)$. Finally from the expression (7.17) we conclude $M, \tau \models \langle\pi^k\rangle\mathbf{L}_\alpha$. ⊠

Lemma 7.5. *Let Σ be a set of actions, $\mathcal{L}_V$ a set of atoms and φ and ψ two DLTL$_{HT}$ formulas over $\mathcal{L}_V$. If $\alpha = \varphi \mathcal{R}^{\pi^*} \psi$, the corresponding $df(\alpha)$ is added to the context and $\mathcal{L}_U = \mathcal{L}_V \cup \{\mathbf{L}_\alpha\}$ (with $\mathbf{L}_\alpha$ a fresh atom), given a DLTL$_{HT}$ interpretation $M = (\sigma, V_h, V_t)$ on $\mathcal{L}_U$, it holds that*

$$M, \tau \models \mathbf{L}_\alpha \textit{ iff } M, \tau \models \alpha$$

for any $\tau \in pref(\sigma)$.

Proof. The proof is analogous to the one shown in Lemma 7.4 but switching the roles of $\wedge$ and $\vee$ and of '$\models$' with '$\not\models$'. ⊠

7.3 Relation to works on planning

Reasoning in dynamic scenarios has also been considered in planning. Planning languages, like PDDL [94], introduces non-monotonicity by assuming false everything which is not defined in a state. Most efficient planners [69, 8] allow specifying domain dependent information in terms of LTL formulas, fact that improves their efficiency but using strategies that are not integrated into the planning language. As a future line of research we aim to develop a planner based on TEL semantics in order to join plan specification, and domain dependent knowledge together.

7.4 Relation to approaches of model checking in ASP

Checking temporal properties on asynchronous systems has been studied in [56]. The method proposed consists in encoding an asynchronous problem into a P/T-nets, a special type of Petri nets, whose inherent concurrency can be exploited in model checking the system. Such nets can be encoded into ASP programs by means of a translation, defined in [56], that captures their possible executions up to n steps. Moreover, such translation can be extended to encode arbitrary LTL formulas into an ASP program in order to check that the net satisfies an LTL formula. The ASP encoding has the form of a splittable temporal logic program and, therefore, it can be used as input for `STeLP` in an straightforward way. Before presenting our translation, we define the concept of P/T-net.

Definition 7.16 (from [56])**.** *A triple $\langle P, T, F \rangle$ is a* net *if $P \cap T = \emptyset$ and $F \subseteq (P \times T) \cup (T \times P)$. The elements of P are called* places *and the elements of T transitions. Places and transitions, usually called* nodes*, are represented in graphical notation by circles and squares respectively. The* flow relation*, F, is represented by arcs. The* preset *of a node is usually denoted by $^\bullet x$ and corresponds to the set $\{y \in P \cup T \mid F(y, x) = 1\}$. Conversely, its* postset *$x^\bullet$ corresponds to $\{y \in P \cup T \mid F(x, y) = 1\}$.* ⊠

We here propose a translation for P/T-nets, which is nothing but a direct translation of the ASP rules proposed in [56] into STeLP syntax.

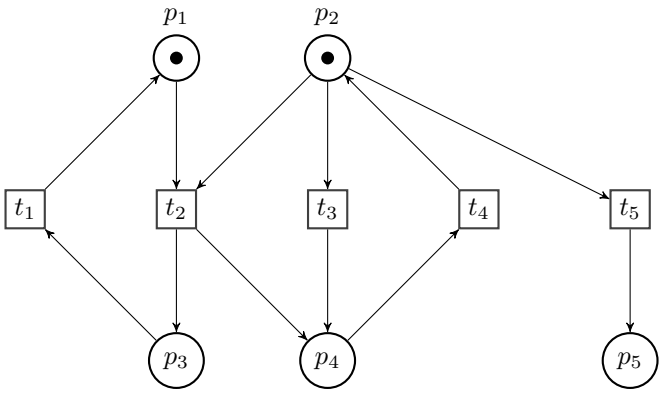


Figure 7.3: Example of a P/T-net from [56]

- For each place p $\in P$ that is marked we add the fact

```
p.
```

- For each translation $t \in T$ we include the TEL formula

$$\Box\left(\left(\left(\left(\bigwedge_{(p_i,t)\in F} p_i\right) \rightarrow t \vee nt\right) \wedge \neg\left(t \wedge nt\right)\right),$$

with p_i an atom in P. This formula means that a transition might be non-deterministically enabled only if all places p such that $(p,t) \in F$, are enabled. This non deterministic behaviour is represented by the disjunction $t \vee nt$, which can lead to models where both atoms t and nt were true at the same time. However, such models are forbidden when adding $\Box\neg\left(t \wedge nt\right)$. This formula corresponds to the following pair of `STeLP` rules:

```
t v nt ::- p1, ..., pn.
::- t, nt.
```

- For each place $p \in P$ add the implication

$$\Box\left(\left(\bigvee_{(t_i,p)\in F} t_i\right) \rightarrow \bigcirc p\right),$$

This formula means that a place will be enabled in the next state if there exists at least one transition is shot in the current one. It can be proved that the previous rule is strongly equivalent to the following conjunction of implications:

$$\left(\bigwedge_{(t_i,p)\in F} \Box\left(t_i \rightarrow \bigcirc p\right) \right)$$

whose translation into STeLP syntax is straightforward. The equivalent representation corresponds to the following program:

```
o p ::- t1.
o p ::- t2.
    ....
    ....
o p ::- tn.
```

- For each place $p \in P$ we add the formula

$$\Box\left(p \wedge \bigwedge_{(t_i,p)\in F} \neg t_i \rightarrow \bigcirc p \right)$$

that emulates the a rule of inertia[3]. This implication can be expressed in STeLP by the rule

```
o p ::- p, t1 .... tn.
```

- For each place $p \in P$ such that $p^\bullet$ contains more than two transitions, we must add the formula

$$\Box\left(\bigwedge_{t_i,t_j\in p^\bullet} \neg\left(t_i \wedge t_j\right) \right),$$

which avoids that a marked place can feed two different transitions. This formula is strongly equivalent to the expression

$$\Diamond\left(\bigvee_{t_i,t_j\in p^\bullet} \left(t_i \wedge t_j\right) \right) \rightarrow \bot$$

expressed as an implication. Since STeLP allows expressing arbitrary LTL formulas in the body of a constraint (see Appendix B), this rule can be directly expressed in our tool as follows:

```
:- pos (( t1 , t2) v (t2, t3) v (t1 v t3) .... ).
```

As an example, let us consider the P/T-net of Figure 7.3. Its STeLP encoding is shown below.

[3] In this example, the rule of inertia for a place p would have the form $\Box\left(p \wedge \neg\bigcirc \sim p \rightarrow \bigcirc p\right)$. However, the truth value of $\bigcirc p$ is determined by the truth value of the transitions in the current moment. Therefore, this rule would behave as inertia does.

```
%% places enabled
p1.       p2.
%% transitions activations
t1 v nt1 ::- p3.
t2 v nt2 ::- p1, p2.
t3 v nt3 ::- p2.
t4 v nt4 ::- p4.
t5 v nt5 ::- p2.
::- t1, nt1.
::- t2, nt2.
::- t3, nt3.
::- t4, nt4.
::- t5, nt5.
%% direct effects
o p1 ::- t1.
o p2 ::- t4.
o p3 ::- t2.
o p4 ::- t2.
o p4 ::- t3.
o p5 ::- t5.
%% inertial laws
o p1 ::- p1, not t2.
o p2 ::- p2, not t2,  not t3, not t5.
o p3 ::- p3, not t1.
o p4 ::- p4, not t4.
o p5 ::- p5.
%% two input transitions of a place
%% cannot be enabled at the same time
:- pos ( (t2, t3) v (t2, t5) v (t3, t5).
```

TEL models correspond to the language accepted by the Büchi automaton of the Figure 7.4, which represents the whole behaviour of the system.

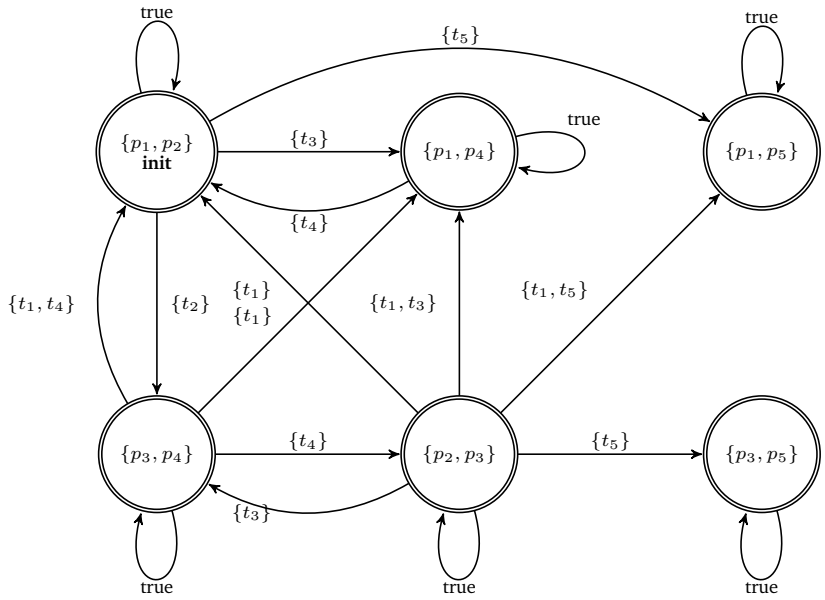


Figure 7.4: Büchi automaton which corresponds to the behaviour of the P/T-net of Figure 7.3

The main advantage of the use of STeLP instead of traditional ASP solvers model checking relies on the fact that it is not necessary to consider bounding model checking. Checking a temporal property φ on STeLP can be performed by adding the formula $\neg\neg\varphi$, to the program and check that the accepting language of the resulting automaton is empty. As an example, let us check whether the P/T-net of Figure 7.3 has deadlocks or not, that is, if the temporal equilibrium models of the net satisfy the LTL formula

$$\neg\neg\Diamond\Box\left((p_1 \vee p_2 \vee p_3 \vee p_4 \vee p_5) \wedge (\neg t_1 \wedge \neg t_2 \wedge \neg t_3 \wedge \neg t_4 \wedge \neg t_5)\right)$$

which corresponds to the following constraint in STeLP:

```
:- always not (always ((p1 v p2 v p3 v p4 v p5),
              (not t1, not t2, not t3, not t4, not t5))).
```

As a result of adding this constraint to our representation we get the automaton of Figure 7.5, which points out that every execution of the net that reaches the state $\{p_1, p_5\}$ enabled would correspond to a state in deadlock.

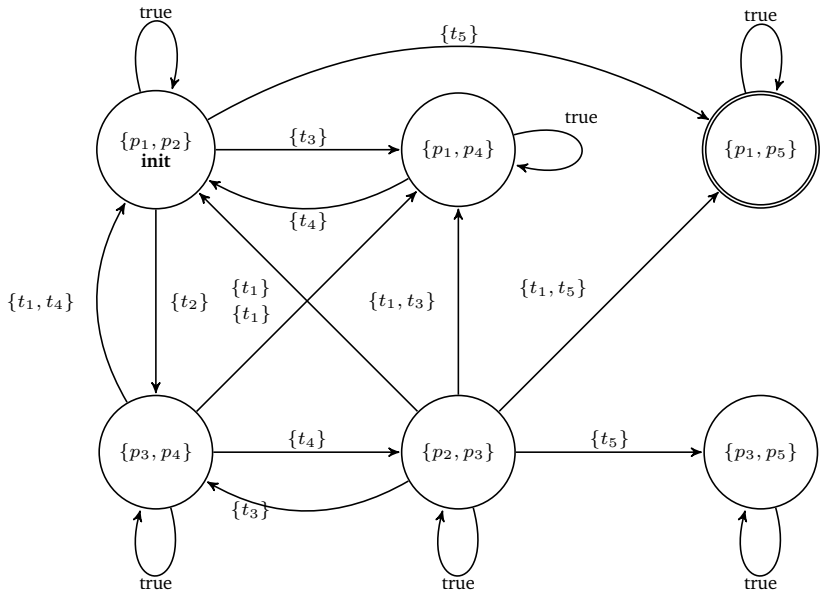


Figure 7.5: All possible executions of the P/T net of Figure 7.3 that lead the net to a deadlock.

7.5 Relation to ER-LTL

ER-LTL [10] is a non-monotonic temporal logic with, as happens with TEL, is also based on LTL, but it is oriented to specify agent's goals in a non-deterministic way. ER-LTL formulas are built from the set of LTL connectives plus a new non-monotonic implication, denoted by $\rightsquigarrow$.

Definition 7.17 (ER-LTL program (from [10]))**.** *Let G, R and P be three disjoint sets of atoms. Let q (the goal atom) be the only atom in G. Let $\langle r \rangle$ (a label) and $\langle p \rangle$ two atoms belonging to R and P respectively. An ER-LTL* rule *is of the form $\langle h : [r]\,(f_1 \rightsquigarrow f_2)$ where $h \in G \cup R$, $r \in R$ and f_1 and f_2 are two ER-LTL formulas. h is referred as the* head*, and $[r](f_1 \rightsquigarrow f_2)$ as the* body *of the rule. Finally an ER-LTL program is a set of the aforementioned rules.*

$\boxtimes$

The main difference with respect to LTL consists in how to interpret the construct $[r](f_1 \rightsquigarrow f_2)$. This construct is read as if f_1 is true then f_2 needs to be satisfied, with the exceptions specified via r, which is defined by means of other rules. For instance we can consider the following representation of the flying bird example:

Example 7.1 (from [10])**.**

$$\langle g : [r_1](bird \rightsquigarrow fly)\rangle \tag{7.19}$$
$$\langle r_1 : [r_2](penguin \rightsquigarrow \neg fly)\rangle \tag{7.20}$$
$$\langle r_1 : [r_3](wounded \rightsquigarrow \top)\rangle \tag{7.21}$$
$$\langle r_2 : [r_4](flying\_penguin \rightsquigarrow fly)\rangle \tag{7.22}$$

⊠

which, intuitively means that birds usually fly but penguins are birds that do not fly an we do not know anything about wounded birds.

The main difference between TEL and ER-LTL comes from the different nature of their semantics. Contrary to TEL semantics, defined in terms of Kripke structures, semantics of ER-LTL programs is defined by means of a translation into a LTL formula, which complicates the task of finding a connection between both formalisms, which is still an open topic. Apparently, TEL seems to be a more general approach to non-monotonic temporal reasoning since no syntactic restriction is imposed.

7.6 Relation to $\mathcal{LAP}_{\rightsquigarrow}$

In this section we consider $\mathcal{LAP}_{\rightsquigarrow}$, a pure modal approach to nonmonotonic reasoning described in [24]. $\mathcal{LAP}_{\rightsquigarrow}$ is based on a monotonic multimodal logic called $\mathcal{LAP}$ plus a dependence relation, $\rightsquigarrow$, between actions and fluents, which is used to solve the frame problem. Formally, $\mathcal{LAP}$ uses two operators $\Box$ (whose corresponding dual operator is denoted by $\Diamond$) and $[\alpha]$ where α corresponds to an action. Logics for each operator ($\Box$ and $[\alpha]$) are S4 and K respectively. On the other hand, $\rightsquigarrow$ states when the execution of an action may make a fluent flip from false to true. For instance, going back to the Yale shooting scenario presented in Section 7.2, there is a dependency relation between the action $shoot$ and fluents $alive$ and $\sim alive$ because after executing $shoot$, some of them may become true. Models of $\mathcal{LAP}_{\rightsquigarrow}$ are those of $\mathcal{LAP}$ satisfying the dependency relation $\rightsquigarrow$.

Despite the fact that both formalisms share decidability and decision procedure, the main difference relies on their underlying monotonic frameworks, while THT considers a linear representation of time, $\mathcal{LAP}$ is branching time, allowing us to talk about possible futures, something which is not possible in our approach. On the other hand, both formalisms also differ in how to solve the frame problem, while TEL avoids it by using rules of inertia, $\mathcal{LAP}_{\rightsquigarrow}$ consider all dependency relations, so all frame actions can be omitted from the $\mathcal{LAP}$ representations.

Chapter 8

Conclusions

In this dissertation we have studied different aspects of *Temporal Equilibrium Logic* (TEL), an innovative combination of a temporal modal logic and a non-monotonic approach. As an overall, this thesis collects a corpus of results that constitutes a significant breakthrough in the knowledge about TEL. The main contributions can be summarised as follows:

(i) We have proved that TEL constitutes an adequate and natural extension of Equilibrium Logic by showing that the syntactic translation proposed by Kamp to encode LTL as a fragment of First Order Logic is also sound for encoding TEL into *Quantified Equilibrium Logic* (QEL) [101].

(ii) We have also provided a translation of TEL into *Infinitary Equilibrium Logic* [55], so that it is possible to apply the idea of *reduct* from that formalism to arbitrary TEL theories.

(iii) We have axiomatised a relevant part of the logic of *Temporal Here-and-There* (THT), the monotonic basis of TEL. In particular, we have provided a set of axioms for operators "always," "eventually" and "next" that properly fixes their semantic behaviour (excepting the linearity of "next" which is left for future work).

(iv) We have shown that THT is a suitable monotonic basis for TEL by proving that the equivalence in this logic is a necessary and sufficient condition for *strong equivalence* of two TEL theories.

(v) We have built a tool, `STELP`, that accepts as input some theory from an expressive syntactic subclass of TEL, called *splittable temporal logic programs*, and computes its temporal equilibrium models in the form of a Büchi automaton. These temporal programs have the informal property that a past formula never depends on references to its future, and practically cover all scenarios represented in the ASP literature.

(vi) We have devised a sound grounding method for removing variables in splittable temporal logic programs. Furthermore, to prove the correctness of this grounding method, we have defined the corresponding first order versions of THT and TEL.

(vii) We have built a second tool, ABSTEM, that allows computing the temporal equilibrium models of any arbitrary theory in the syntax of (proposional) LTL, removing in this way any syntactic restriction. We have also used this tool for checking different types of equivalence (LTL equivalence, TEL equivalence and strong equivalence) between a pair of theories and showing information (countermodels) when the theories are not equivalent.

(viii) Finally, we have compared TEL to other non-monotonic approaches for representing temporal scenarios. Using the translation into Infinitary Equilibrium Logic, we have been able to prove that the semantics of TEMPLOG [11], a temporal logic programming approach without default negation, actually coincides with the result obtained by TEL for that syntactic class. We have also extended the comparison to Temporal Answer Sets [51] made in [4] by showing a partial translation of the former into TEL via the introduction of auxiliary atoms.

The main difficulty of this research program has been related to the variety of fundamental results and methods that have been required from the two fields, temporal modal logic and NMR. For instance, some theoretical contributions, like (iv), have been pursued since the first steps of this thesis but only solved very recently and after a deeper study on previous related work from different areas. Similarly, (iii) constitutes recent, unpublished research developed during my stay at IRIT, University of Toulouse with the unvaluable collaboration and guidance of Philippe Balbiani. The implementation of different tools has also required an important effort on combination of ASP solvers and software for LTL and automata construction.

There are many aspects in which this work can be extended or completed in the future. We outline next some of the most salient ones:

- A first obvious topic is completing the work on axiomatisation of THT to prove that it fully characterises linear narratives: until now, we have been able to prove that the necessity operator covers the transitive closure of "next," and that there always exists an accessible world, but not its uniqueness. This axiomatisation can also be extended to cope with the "until" and "release" operators.

- Another interesting theoretical question is whether one of these operators is representable in terms of the other or not. It is known that the LTL definitions of "until" in terms of "release" or vice versa are not applicable in the case of THT (mainly because double negation cannot be removed in the general case). However, disjunction in Here-and-There (HT) is expressible in terms of implication and conjunction and, similarly, in Quantified HT the existential quantifier is expressible in terms of the universal one (again, with implication and conjunction). It remains to know whether "until" follows an analogous relation with respect to "release."

- Although we have proved that TEL can be translated into Quantified Equilibrium Logic following Kamp's translation of LTL into Monadic First

Order Logic, $MFO(<)$, we ignore whether the other direction of Kamp's theorem also holds in this case or not. Namely, we ignore whether any theory in Monadic Quantified Equilibrium Logic for a linear order relation $<$ can be represented in TEL.

- Following similar steps as those done in TEL, other hybrid approaches can be explored. For instance, [4] has considered the combination of Dynamic LTL with Equilibrium Logic. Similarly, other temporal approaches can be treated in an analogous way, such as CTL, CTL*, Dynamic Logic or μ-calculus. Another possibility is considering other NMR or Logic Programming semantics such as Well-Founded Semantics [118] or Completion [27].

- Stable models of a propositional theory can be captured by a classical formula by the use of loop formulas. Unfortunately, this is not always possible when dealing with Quantified Equilibrium Logic. In [70], the first-order stable models of several syntactic subclases of QEL are proved to be captured by a first order sentece. In our case, regarding TEL as a fragment of QEL, we wonder whether the set of temporal equilibrium models of a formula φ is first-order (and consequently LTL) representable[1].

- As for implementation, a possibility that has been initially explored but eventually left for future work is, instead of applying automata-based methods, using the tool `TSPASS` [41] (based on temporal resolution) as a back-end. This alternative could be compared to the current implementations and an efficiency assessment could be made depending on different types of input theories.

- Finally, an interesting line of research is exploiting the obtained results for their application to action languages, providing translations of current formalisms related to ASP such as $\mathcal{ALM}$ [62] or MAD [80]. Similarly, another area where the current background on TEL can be applied is AI Planning, especially for comparing with planners that use temporal constraints for heuristic control rules.

[1] A first attempt to give an answer to this question was published in [21] but an error in the proof makes the answer is still unknown

Bibliography

[1] F. Aguado, P. Cabalar, M. Diéguez, G. Pérez, and C. Vidal. Paving the way for temporal grounding. In *Proc. of the 28th International Conference on Logic Programming (ICLP'12)*, pages 290–300, Budapest, Hungary, 2012. Cited on page 81.

[2] F. Aguado, P. Cabalar, G. Pérez, and C. Vidal. Strongly Equivalent Temporal Logic Programs. In *Proc. of the 11th European Conference on Logics in Artificial Intelligence (JELIA'08)*, pages 8–20, Dresden, Germany, 2008. Cited on pages 3, 45, 107, 109, and 177.

[3] F. Aguado, P. Cabalar, G. Pérez, and C. Vidal. Loop Formulas for Splitable Temporal Logic Programs. In *Proc. of the 11th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'11)*, pages 80–92, Vancouver, Canada, 2011. Cited on pages 86 and 163.

[4] F. Aguado, G. Pérez, and C. Vidal. Integrating Temporal Extensions of Answer Set Programming. In *Proc. of the 12th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'13)*, pages 23–35, Corunna, Spain, 2013. Cited on pages 121, 123, 124, 125, 140, 141, and 179.

[5] M. Alviano, F. Calimeri, G. Charwat, M. Dao-Tran, C. Dodaro, G. Ianni, T. Krennwallner, M. Kronegger, J. Oetsch, A. Pfandler, J. Pührer, C. Redl, F. Ricca, P. Schneider, M. Schwengerer, L. Katharina Spendier, J. Peter Wallner, and G. Xiao. The fourth answer set programming competition: Preliminary report. In *Proc. of the 12th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'13)*, pages 42–53, Corunna, Spain, 2013. Cited on page 6.

[6] A. Artale and E. Franconi. A survey of temporal extensions of description logics. *Annals of Mathematics and Artificial Intelligence*, 30:171–210, 2000. Cited on page 4.

[7] F. Bacchus and F. Kabanza. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence*, 116:123–191, 2000. Cited on page 4.

[8] J. A. Baier and S. A. McIlraith. Planning with temporally extended goals using heuristic search. In *Proc. of the 16th International Conference on*

Automated Planning and Scheduling (ICAPS'06), pages 342–345, Cumbria, UK, 2006. Cited on pages 4 and 131.

[9] P. Balbiani and D. Vakarelov. PDL with intersection of programs: a complete axiomatization. *Journal of Applied Non-Classical Logics*, 13:231–276, 2003. Cited on page 4.

[10] C. Baral and J. Zhao. Non-monotonic temporal logics that facilitate elaboration tolerant revision of goals. In *Proc. of the 23th AAAI Conference on Artificial Intelligence (AAAI'08)*, pages 406–411, Chicago, Illinois, USA, 2008. Cited on pages 41, 136, and 137.

[11] M. Baudinet. A simple proof of the completeness of temporal logic programming. In L. Fariñas del Cerro and M. Penttonen, editors, *Intensional Logics for Programming*, pages 51–83. Clarendon Press, 1992. Cited on pages 115, 116, 117, 140, and 178.

[12] N. Bidoit and C. Froidevaux. Minimalism subsumes default logic and circumscription in stratified logic programming. In *Proc. of the IEEE Symposium on Logic in Computer Science (LICS'87)*, pages 89–97, Ithaca, New York, USA, 1987. Cited on pages 6 and 11.

[13] L. Bozzelli and D. Pearce. On the complexity of temporal equilibrium logic satisfiability. Unpublished draft. Cited on page 103.

[14] G. Brewka, T. Eiter, and M. Truszczyński. Answer Set Programming at a Glance. *Communications of the ACM*, 54(12):92–103, 2011. Cited on pages 7, 11, and 12.

[15] A. Bria, W. Faber, and N. Leone. Normal Form Nested Programs. In *Proc. of the 11th European Conference on Logics in Artificial Intelligence (JELIA'08)*, pages 76–88, Dresden, Germany, 2008. Cited on page 22.

[16] J. R. Büchi. On a Decision Method in Restricted Second Order Arithmetic. In *International Congress on Logic, Methodology, and Philosophy of Science*, pages 1–11, 1962. Cited on pages 38, 40, and 81.

[17] P. Cabalar. A Normal Form for Linear Temporal Equilibrium Logic. In *Proc. of the 12th European Conference on Logics in Artificial Intelligence (JELIA'10)*, pages 64–76, Helsinki, Findland, 2010. Cited on pages 3, 43, 44, and 177.

[18] P. Cabalar and S. Demri. Automata-Based Computation of Temporal Equilibrium Models. In *Proc. of the 21st International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR'11)*, pages 57–72, Odense, Denmark, 2011. Cited on pages 44, 45, 47, 48, 81, 100, 103, 107, 111, 171, and 172.

[19] P. Cabalar and M. Diéguez. STeLP - A Tool for Temporal Answer Set Programming. In *Proc. of the 11th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'11)*, pages 370–375, Vancouver, Canada, 2011. Cited on page 97.

[20] P. Cabalar and M. Diéguez. Strong equivalence of non-monotonic temporal theories. In *Proc. of the 14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14)*, Vienna, Austria, 2014. Cited on pages 81, 107, and 171.

[21] P. Cabalar and M. Diéguez. Temporal Stable Models are LTL-representable. In *Proc. of 7th Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP'14)*, Vienna, Austria, 2014. Cited on page 141.

[22] P. Cabalar and P. Ferraris. Propositional theories are strongly equivalent to logic programs. *Theory and Practice of Logic Programming*, 7(6):745–759, 2007. Cited on pages 29 and 30.

[23] P. Cabalar and G. Pérez. Temporal Equilibrium Logic: A First Approach. In *Proc. of the 11th International Conference on Computer Aided Systems Theory (EUROCAST'07)*, page 241–248, Las Palmas de Gran Canaria, Spain, 2007. Cited on pages 2, 3, 46, 176, and 177.

[24] M. A. Castilho, O. Gasquet, and A. Herzig. Formalizing action and change in modal logic I: the frame problem. *Journal of Logic and Computation*, 9(5):701–735, 1999. Cited on pages 6 and 137.

[25] A. V. Chagrov and M. Zakharyaschev. *Modal Logic*, volume 35 of *Oxford Logic Guides*. Oxford University Press, 1997. Cited on pages 54, 56, 57, and 63.

[26] Y. Chen, F. Lin, and L. Li. SELP - a system for studying strong equivalence between logic programs. In *Proc. of the 8th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'05)*, pages 442–446, Diamante, Italy, 2005. Cited on page 31.

[27] K. L. Clark. Negation as Failure. In *Logic and Databases*, pages 293–322. Plenum Press, 1978. Cited on pages 15, 141, and 180.

[28] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, 1986. Cited on page 4.

[29] J. Couvreur. On-the-fly verification of temporal logic. In *Proc. of the World Congress on Formal Methods in the Development of Computing Systems (FM'99)*, pages 253–271, Toulouse, France, 1999. Cited on page 103.

[30] D. Van Dalen. Intuitionistic logic. In *Handbook of Philosophical Logic*, volume 166, pages 225–339. Springer Netherlands, 1986. Cited on page 32.

[31] A. Duret-Lutz. LTL Translation Improvements in Spot 1.0. *International Journal on Critical Computer-Based Systems*, 5(1/2):31–54, 2014. Cited on page 100.

[32] T. Eiter, W. Faber, N. Leone, G. Pfeifer, and A. Polleres. The DLV^{K} Planning System: Progress Report. In *Proc. of the 8th European Conference on Logics in Artificial Intelligence (JELIA'02)*, pages 541–544, Cosenza, Italy, 2002. Cited on page 7.

[33] E. A. Emerson and J. Y. Halpern. "sometimes" and "not never" revisited: on branching versus linear time temporal logic. *Journal of the ACM*, 33(1):151–178, 1986. Cited on page 4.

[34] E. Erdem and V. Lifschitz. Tight logic programs. *Theory and Practice of Logic Programming*, 3(4-5):499–518, 2003. Cited on page 16.

[35] F. Fages. Consistency of clark's completion and existence of stable models. *Journal of Methods of Logic in Computer Science*, 1(1):51–60, 1994. Cited on pages 16 and 47.

[36] P. Ferraris. Answer Sets for Propositional Theories. In *Proc. of 8th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'05)*, pages 119–131, Diamante, Italy, 2005. Cited on pages 17 and 46.

[37] P. Ferraris, J. Lee, and V. Lifschitz. A Generalization of the Lin-Zhao Theorem. *Annals of Mathematics and Artificial Intelligence*, 47(1-2):79–101, 2006. Cited on pages 17, 18, 19, 84, and 85.

[38] P. Ferraris, J. Lee, and V. Lifschitz. A New Perspective on Stable Models. In *Proc. of 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 372–379, Hyderabad, India, 2007. Cited on pages 7, 23, 24, 25, and 33.

[39] M. Fischer and R. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18:194–211, 1979. Cited on page 4.

[40] M. Fisher. Temporal semantics for concurrent metatem. *Journal of Symbolic Computation*, 22(5/6):627–648, 1996. Cited on page 4.

[41] M. Fisher, C. Dixon, and M. Peim. Clausal temporal resolution. *ACM Transactions on Computational Logic*, 2(1):12–56, 2001. Cited on pages 141 and 180.

[42] D. Gabbay, I. Hodkinson, and M. A. Reynolds. *Temporal Logic: Mathematical Foundations and Computational Aspects Volume 1*. Number 28 in Oxford Logic guides. Clarendon Press, 1994. Cited on pages 3 and 38.

[43] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the Temporal Analysis of Fairness. In *Proc. of the 7th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'80)*, pages 163–173, Las Vegas, Nevada, USA, 1980. Cited on pages 3 and 38.

[44] M. Gebser, T. Grote, and T. Schaub. Coala: A Compiler from Action Languages to ASP. In *Proc. of the 12th European Conference Logics in Artificial Intelligence (JELIA'10)*, pages 360–364, Helsinki, Finland, 2010. Cited on page 7.

[45] M. Gebser, B. Kaufmann, and T. Schaub. Conflict-driven answer set solving: From theory to practice. *Artificial Intelligence*, 187-188:52–89, 2012. Cited on page 6.

[46] M. Gebser, T. Schaub, and S. Thiele. Gringo: A new grounder for answer set programming. In *Proc. of the 9th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'07)*, pages 266–271, Tempe, Arizona, USA, 2007. Cited on pages 19 and 22.

[47] M. Gelfond and D. Inclezan. Yet Another Modular Action Language. In *Proc. of the LPNMR-09 Workshop on Software Engineering for Answer Set Programming, (SEA'09)*, pages 64–78, Potsdam, Germany, 2009. Cited on page 7.

[48] M. Gelfond and V. Lifschitz. The Stable Model Semantics For Logic Programming. In *Proc. of the 5th International Conference on Logic Programming (ICLP'88)*, page 1070–1080, Seattle, Washington, USA, 1988. Cited on pages 2, 6, 11, 12, 13, 25, 121, and 176.

[49] M. Gelfond and V. Lifschitz. Action Languages. *Electronic Transactions Artificial Intelligence*, 2:193–210, 1998. Cited on page 7.

[50] G. De Giacomo, Y. Lespérance, and F. Patrizi. Bounded situation calculus action theories and decidable verification. In *Proc. of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR'12)*, pages 467–477, Rome, Italy, 2012. Cited on page 6.

[51] L. Giordano, A. Martelli, and D. Theseider Dupré. Reasoning about actions with Temporal Answer Sets. *Theory and Practice of Logic Programming*, 13:201–225, 3 2013. Cited on pages 6, 8, 9, 41, 119, 120, 121, 140, and 179.

[52] K. Gödel. Zum intuitionistischen Aussagenkalkül. *Anzeiger der Akademie der Wissenschaften Wien, mathematisch, naturwissenschaftliche Klasse*, 69:65–66, 1932. Cited on page 28.

[53] R. Goldblatt. *Logics of Time and Computation*. Number 7 in CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, California, 2 edition, 1992. Cited on pages 4, 53, 54, 62, 69, 71, and 80.

[54] S. Hanks and D. McDermott. Nonmonotonic logic and temporal projection. *Artificial Intelligence*, 33(3):379–412, 1987. Cited on page 122.

[55] A. Harrison, V. Lifschitz, D. Pearce, and A. Valverde. Infinitary Equilibrium Logic. In *Proc. of 7th Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP'14)*, Vienna, Austria, 2014. Cited on pages 8, 33, 34, 50, 119, 139, and 178.

[56] K. Heljanko and I. Niemelä. Bounded LTL model checking with stable models. *Theory and Practice of Logic Programming*, 3(4-5):519–550, 2003. Cited on pages 131 and 132.

[57] A. Heyting. *Die formalen Regeln der intuitionistischen Logik*. Sitzungsberichte der Preussischen Akademie der Wissenschaften. Physikalisch-mathematische Klasse. Deütsche Akademie der Wissenschaften zu Berlin, Mathematisch-Naturwissenschaftliche Klasse, 1930. Cited on pages 2, 27, 122, and 177.

[58] I Hodkinson. Expressive completeness of until and since over dedekind complete linear time. *Modal logic and process algebra*, 53:171–185, 1995. Cited on page 3.

[59] G. J. Holzmann. *The SPIN Model Checker - primer and reference manual*. Addison-Wesley, 2004. Cited on page 4.

[60] T. Hosoi. The Axiomatization of the Intermediate Propositional Systems S_2 of Gödel. *Journal of the Faculty of Science of the University of Tokyo*, 13(2):183–187, 1966. Cited on pages 53 and 54.

[61] M. Huth and M. Ryan. *Logic in Computer Science: Modelling and Reasoning About Systems*. Cambridge University Press, 2004. Cited on page 166.

[62] D. Inclezan and M. Gelfond. Representing biological processes in modular action language ALM. In *Logical Formalizations of Commonsense Reasoning, AAAI Spring Symposium*, Stanford, California, USA, 2011. Cited on pages 141 and 180.

[63] H. Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, California, USA, 1968. Cited on pages 1, 2, 3, 37, 38, 175, and 176.

[64] C. R. Karp. *Languages with expressions of infinite length*. Studies in logic and the foundations of mathematics. North-Holland Publication Company, 1964. Cited on page 33.

[65] H. A. Kautz and B. Selman. Planning as satisfiability. In *Proc. of the European Conf. on Artificial Intelligence (ECAI'92)*, pages 359–363, 1992. Cited on page 8.

[66] R. Kowalski and M. Sergot. A logic-based calculus of events. *New Generation Computing*, 4:67–95, 1986. Cited on page 5.

[67] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27(3):333–354, 1983. Cited on page 4.

[68] S. A. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963. Cited on page 35.

[69] J. Kvarnström and P. Doherty. Talplanner: A temporal logic based forward chaining planner. *Annals of Mathematics and Artificial Intelligence*, 30(1-4):119–169, 2000. Cited on pages 4 and 131.

[70] J. Lee and Y. Meng. On loop formulas with variables. In *11^{11} International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, pages 444–453, Sydney, Australia, 2008. Cited on page 141.

[71] J. Lee and R. Palla. System f2lp - computing answer sets of first-order formulas. In *Proc. of the 10^{th} International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'09)*, pages 515–521, Potsdam, Germany, 2009. Cited on page 7.

[72] J. Lee and R. Palla. Reformulating the situation calculus and the event calculus in the general theory of stable models and in answer set programming. *Journal of Artificial Intelligence Research (JAIR)*, 43:571–620, 2012. Cited on page 7.

[73] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. The DLV system for knowledge representation and reasoning. *ACM Transactions on Computational Logic*, 7:499–562, 2006. Cited on pages 6, 19, and 22.

[74] L. Libkin. *Elements of Finite Model Theory*. Texts in Theoretical Computer Science. An Eatcs Series. SpringerVerlag, 2004. Cited on page 26.

[75] V. Lifschitz. Circumscription. In *Handbook of Logic in AI and Logic Programming*, volume 3, pages 297–352. Oxford University Press, 1994. Cited on page 24.

[76] V. Lifschitz. Twelve definitions of a stable model. In *Proc. of the 24th Intl. Conf. on Logic Programming (ICLP'08)*, pages 37–51, Udine, Italy, 2008. Cited on page 7.

[77] V. Lifschitz. Thirteen Definitions of a Stable Model. In *Fields of Logic and Computation: Essays Dedicated to Yuri Gurevich on the Occasion of his 70^{th} Birthday*, pages 488–503, 2010. Cited on page 23.

[78] V. Lifschitz, D. Pearce, and A. Valverde. Strongly Equivalent Logic Programs. *ACM Transactions on Computational Logic*, 2(4):526–541, 2001. Cited on pages 7 and 31.

[79] V. Lifschitz, D. Pearce, and A. Valverde. A Characterization of Strong Equivalence for Logic Programs with Variables. In *Proc. of the 9^{th} International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'07)*, pages 188–200, Tempe, Arizona, USA, 2007. Cited on pages 31, 33, 107, 109, and 110.

[80] V. Lifschitz and W. Ren. A modular action description language. In *Proc. of the 21st National Conference on Artificial Intelligence (AAAI'06)*, pages 853–859, Boston, Massachusetts, USA, 2006. Cited on pages 141 and 180.

[81] V. Lifschitz, L. R. Tang, and H. Turner. Nested expressions in logic programs. *Annals of Mathematics and Artificial Intelligence*, 25:369–389, 1999. Cited on page 17.

[82] V. Lifschitz and H. Turner. Splitting a Logic Program. In *Proc. of the 11th International Conference on Logic Programming (ICLP'94)*, pages 23–37, Santa Marherita Ligure, Italy, 1994. Cited on pages 14, 15, and 84.

[83] F. Lin. Reducing Strong Equivalence of Logic Programs to Entailment in Classical Propositional Logic. In *Proc. of the 8th Intl. Conf. on Principles and Knowledge Representation and Reasoning (KR'02)*, pages 170–176, Toulouse, France, 2002. Cited on page 31.

[84] F. Lin and Y. Zhao. ASSAT: Computing Answer Sets of a Logic Program by SAT Solvers. In *Artificial Intelligence*, pages 112–117, 2002. Cited on pages 17 and 18.

[85] J. Lukasiewicz. Die logik und das grundlagenproblem. *Les Entreties de Zürich sur les Fondaments et la Méthode des Sciences Mathématiques*, 12(6-9):82–100, 1938. Cited on page 53.

[86] C. Lutz, F. Wolter, and M. Zakharyaschev. Temporal description logics: A survey. In *Proc. of the 15th International Symposium on Temporal Representation and Reasoning (TIME '08)*, pages 3–14, Montreal, Canada, 2008. IEEE Computer Society. Cited on page 4.

[87] V. Marek and M. Truszczyński. *Nonmonotonic logic: context-dependent reasoning*. Artificial intelligence. Springer, 1993. Cited on pages 2 and 176.

[88] V. Marek and M. Truszczyński. *Stable models and an alternative logic programming paradigm*, pages 169–181. Springer-Verlag, 1999. Cited on page 6.

[89] J. McCarthy. Programs with commonsense. In *Proc. of the Teddington Conference on the Mechanization of Thought Processes*, pages 75–91, London, UK, 1959. Cited on page 5.

[90] J. McCarthy. Circumscription: A Form of Non-Monotonic Reasoning. *Artificial Intelligence*, 13:27–39, 1980. Cited on pages 5 and 24.

[91] J. McCarthy. Modality, si! modal logic, no! *Studia Logica*, 59(1):29–32, 1997. Cited on page 6.

[92] J. McCarthy. Elaboration tolerance. In *Proc. of the 4th Symposium on Logical Formalizations of Commonsense Reasoning (Common Sense 98)*, pages 198–217, London, UK, 1998. Updated version at `http://www-formal.stanford.edu/jmc/elaboration.ps`. Cited on page 5.

[93] J. McCarthy and P. J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence Journal*, 4:463–502, 1969. Cited on page 5.

[94] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL - The Planning Domain Definition Language. Technical Report TR-98-003, Yale Center for Computational Vision and Control, 1998. Cited on page 131.

[95] R. McNaughton and S. A. Papert. *Counter-Free Automata (M.I.T. Research Monograph No. 65)*. The MIT Press, 1971. Cited on page 40.

[96] I. Niemelä. Logic Programs with Stable Model Semantics as a Constraint Programming Paradigm. *Annals of Mathematics and Artificial Intelligence*, 25(3-4):241–273, 1999. Cited on page 6.

[97] M. Nogueira, M. Balduccini, M. Gelfond, R. Watson, and M. Barry. An A-Prolog decision support system for the space shuttle. In *AAAI Spring Symposium*, 2001. Cited on page 7.

[98] D. Pearce. A New Logical Characterisation of Stable Models and Answer Sets. In *Proc. of Non-Monotonic Extensions of Logic Programming (NMELP'96)*, pages 57–70, Bad Honnef, Germany, 1996. Cited on pages 2, 27, and 176.

[99] D. Pearce. Equilibrium Logic. *Annals of Mathematics and Artificial Intelligence*, 47(1-2):3–41, 2006. Cited on pages 7, 27, 28, 29, and 88.

[100] D. Pearce, H. Tompits, and S. Woltran. Encodings for Equilibrium Logic and Logic Programs with Nested Expressions. In *Proc. of the 10th Portuguese Conference on Artificial Intelligence (EPIA'01)*, pages 306–320, Porto, Portugal, 2001. Cited on pages 24 and 30.

[101] D. Pearce and A. Valverde. Quantified Equilibrium Logic and Foundations for Answer Set Programs. In *Proc. of the 24th International Conference on Logic Programming (ICLP'08)*, pages 546–560, Udine, Italy, 2008. Cited on pages 7, 8, 31, 32, 48, 139, and 178.

[102] A. Pnueli. The Temporal Logic of Programs. In *Proc. of the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57, Providence, Rhode Island, USA, 1977. Cited on page 4.

[103] A. Prior. *Past, Present and Future*. Oxford books. Oxford University Press, 1967. Cited on pages 2, 3, and 176.

[104] A. Rabinovich. A Proof of Kamp's Theorem. *Logical Methods in Computer Science*, 10(1), 2014. Cited on pages 3 and 38.

[105] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13:81–132, 1980. Cited on page 5.

[106] R. Reiter. Non-monotonic logic I. *D. V. McDermott and J. Doyle*, 13:41–72, 1980. Cited on page 5.

[107] S. Safra. *Complexity of Automata on Infinite Objects*. PhD thesis, Weizmann Institute of Science, Rehovot, Israel, 1989. Cited on pages 102 and 103.

[108] E. Sandewall. *Features and Fluents*. Oxford Logic Guides. Clarendon Press, 1995. Cited on page 5.

[109] D. Scott and A. Tarski. The Sentential Calculus With Infinitely Long Expressions. *Colloquium Mathematicae*, 6(1):165–170, 1958. Cited on page 33.

[110] A. K. Simpson. *The Proof Theory and Semantics of Intuitionistic Modal Logic*. PhD thesis, University of Edinburgh, UK, 1994. Cited on page 62.

[111] A. P. Sistla and E. M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985. Cited on page 4.

[112] A. P. Sistla, M. Y. Vardi, and P. Wolper. The complementation problem for büchi automata with applications to temporal logic. In *Proc. of the 12th International Colloquium on Automata, Languages and Programming*, volume 194, pages 465–474, Nafplion, Greece, 1985. Cited on page 102.

[113] J. F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing Company, 2000. Cited on page 5.

[114] R. Streett. Propositional dynamic logic of looping and converse is elementary decidable. *Information and Control*, 54:121–141, 1982. Cited on page 4.

[115] M. Truszczyński. Connecting First-Order ASP and the Logic FO(ID) through Reducts. In *Correct Reasoning - Essays on Logic-Based AI in Honour of Vladimir Lifschitz*, volume 7265 of *Lecture Notes in Computer Science*, pages 543–559. Springer, 2012. Cited on pages 33, 52, and 119.

[116] A. Valverde. tabeql: A Tableau Based Suite for Equilibrium Logic. In *Proc. of the 9th European Conference on Logics in Artificial Intelligence (JELIA'04)*, pages 734–737, Lisbon, Portugal, 2004. Cited on page 31.

[117] M. H. van Emden and R. A. Kowalski. The Semantics of Predicate Logic as a Programming Language. *Journal of the ACM*, 23(4):733–742, 1976. Cited on pages 12 and 13.

[118] A. van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991. Cited on pages 141 and 180.

[119] M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logics of programs. *Journal of Computer and System Sciences*, 2(32):183–219, 1986. Cited on page 38.

[120] M. Y. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115:1–37, 1994. Cited on pages 4, 100, and 102.

[121] P. Wolper. The tableau method for temporal logic: An overview. *Logique Et Analyse*, 28(110-111):119–136, 1985. Cited on pages 4 and 53.

Appendix A

Proofs of auxiliary lemmas of Chapter 4

A.1 System properties

Proof of Proposition 4.1 (iii). Since $\Gamma_H \subseteq \Gamma_T$, $\alpha \to \beta$ also belongs to Γ_T. Finally, from Proposition 4.1 (ii) we get $\alpha \in \Delta_H$ or $\beta \in \Gamma_H$ and $\alpha \in \Delta_T$ or $\beta \in \Gamma_T$ too. ⊠

Proof of Proposition 4.1 (iv). Since Axiom (1) from Table 4.1, $\beta \to (\alpha \to \beta)$, and 4.1 (ii) we conclude that $\beta \in \Delta_H$.

On the other hand, from Axiom 6 from Table 4.1 we conclude that $\alpha \to \alpha \vee \beta$, which together with $\beta \in \Delta_H$, derives either $\alpha \in \Gamma_H$ or Δ_H. while in the former case we reach the conclusion, in the latter we can use Definition 4.4 to derive both $\alpha \in \Gamma_T$ and $\beta \in \Delta_T$, which is the second part of the conclusion. ⊠

Proof of Proposition 4.1 (v). From left to right, from Axiom (1) from Table 4.1 we derive that $\beta \to (\alpha \to \beta) \in \Gamma_T$. By saturation we derive that $\beta \in \Delta_T$. Moreover, since $\alpha \to \beta \in \Delta_T$ then $\alpha \to \beta \in \Delta_H$ and, from Proposition 4.1 (iv) and $\Gamma_H \subseteq \Gamma_T$, we derive that $\alpha \in \Gamma_T$.

From right to left, assume $\alpha \in \Gamma_T$, $\beta \in \Delta_T$ and $\alpha \to \beta \notin \Delta_T$. In this case $\alpha \to \beta \in \Gamma_T$ and, by means of Modus Ponens we derive $\beta \in \Gamma_T$ and this contradicts the consistency of the tableau (Γ_T, Δ_T).

⊠

Proof of Proposition 4.1 (vi).

From left to right, since $\neg\neg\alpha \in \Gamma_H$ and $\Gamma_H \subseteq \Gamma_T$ we get $\neg\neg\alpha \in \Gamma_T$ and, since the tableau is saturated, $\neg\alpha \in \Delta_T$. Finally, from Proposition 4.1 (v) we conclude $\alpha \in \Gamma_T$.

From right to left, assume that $\neg\neg\alpha \notin \Gamma_H$ therefore, it belongs to Δ_H. From Proposition 4.1 (iv) and the property of $\Gamma_H \subseteq \Gamma_T$ we conclude that $\neg\alpha \in \Gamma_T$, which contradicts the consistency of the tableau (Γ_T, Δ_T). ⊠

A.2 Consistency of tableaux

Lemma A.1. *Let $t = (\Gamma^t, \Delta^t)$ be a consistent tableau and φ a formula, if $\neg\neg\Box\varphi \in \Delta^t$ then the tableau $u = (\Box\Gamma^t, \Diamond\Delta \cup \{\neg\neg\varphi^t\})$ is consistent.*

Proof. Assume that u is not consistent, then there exists $\phi_1, \cdots, \phi_m \in \Box\Gamma^t$ and $\chi_1, \cdots, \chi_n \in \Diamond\Delta^t$ such that

$$\phi_1 \wedge \cdots \wedge \phi_m \rightarrow \chi_1 \vee \cdots \vee \chi_n \vee \neg\neg\varphi$$

is satisfied, so if we apply the Rule (13) from Table 4.1, we derive

$$\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \neg\neg\Box\varphi$$

whose antecedent and consequent belong to Γ^t and Δ^t respectively. Hence, it would contradict the consistency of t. ⊠

Lemma A.2. *Let $t = (\Gamma^t, \Delta^t)$ be a consistent tableau and φ a formula. If $\Diamond\varphi \in \Gamma^t$ then the tableau $u = (\Box\Gamma^t \cup \{\varphi\}, \Diamond\Delta^t)$ is consistent.*

Proof. Assume that u is not consistent, so there exists $\phi_1, \cdots, \phi_m \in \Box\Gamma^t$ and $\chi_1, \cdots, \chi_n \in \Diamond\Delta^t$ such that

$$\phi_1 \wedge \cdots \wedge \phi_m \wedge \varphi \rightarrow \chi_1 \vee \cdots \vee \chi_n$$

is satisfied. However, after applying the Rule (14) from Table 4.1 we obtain the rule

$$\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \wedge \Diamond\varphi \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n$$

whose antecedent and consequent belong to Γ^t and Δ^t. This implies that t is not consistent, which contradicts our initial assumption. ⊠

Lemma A.3. *Let $t = (\Gamma^t, \Delta^t)$ be a consistent tableau and φ a formula. If $\neg\neg\Diamond\varphi \in \Gamma^t$ then the tableau $u = (\Box\Gamma^t \cup \{\neg\neg\varphi\}, \Diamond\Delta^t)$ is consistent.*

Proof. Assume that u is not consistent, then there exists $\phi_1, \cdots, \phi_m \in \Box\Gamma^t$ and $\chi_1, \cdots, \chi_n \in \Diamond\Delta^t$ such that

$$\phi_1 \wedge \cdots \wedge \phi_m \wedge \varphi \wedge \neg\neg\varphi \rightarrow \chi_1 \vee \cdots \vee \chi_n$$

is satisfied. However, after applying the Rule (15) from Table 4.1 we obtain the rule

$$\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \wedge \neg\neg\Diamond\varphi \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n$$

whose antecedent and consequent belong to Γ^t and Δ^t, contradicting the consistency of t. ⊠

Lemma A.4. *Let $t = (\Gamma^t, \Delta^t)$ be a consistent tableau and φ a formula. If $\Box\varphi \in \Delta^t$ then the tableau $u = (\Box\Gamma^t, \Diamond\Delta^t \cup \{\varphi\})$ is consistent*

Proof. Assume that u is not consistent. Therefore there exists $\phi_1, \cdots, \phi_m \in \Box\Gamma^t$ and $\chi_1, \cdots, \chi_n \in \Diamond\Delta^t$ such that

$$\phi_1 \wedge \cdots \wedge \phi_m \rightarrow \chi_1 \vee \cdots \vee \chi_n \vee \varphi,$$

is satisfied. However, after applying the Rule (12) from Table 4.1 we obtain

$$\Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \Box\varphi.$$

However, this implies that t is not consistent and, therefore, a contradiction. ⊠

Corollary A.1. *Let $t = (\Gamma^t, \Delta^t)$ be a consistent tableau and φ a formula. If $\hat{\bigcirc}\varphi \in \Gamma^t$ then the tableau $u = (\bigcirc\Gamma^t \cup \{\varphi\}, \hat{\bigcirc}\Delta^t)$ is consistent.*

Proof. The same proof as for Lemma A.4. ⊠

Corollary A.2. *Let $t = (\Gamma^t, \Delta^t)$ be a consistent tableau and φ a formula. If $\bigcirc\varphi \in \Delta^t$ then the tableau $u = (\bigcirc\Gamma^t, \hat{\bigcirc}\Delta^t \cup \{\varphi\})$ is consistent*

Proof. The same proof as for Lemma A.2. ⊠

A.3 Soundness

In this section we check that the axiomatic system shown in Table 4.1 is sound. The proof of soundness would consist in checking that all axioms are valid and inference rules preserve validity, always with respect to their corresponding class of frames.

Lemma A.5. *The following inference rule:*

$$\frac{\phi_1 \wedge \cdots \wedge \phi_n \rightarrow \chi_1 \vee \cdots \vee \chi_n}{\phi_1 \vee (\phi_1 \rightarrow \chi_1) \vee \cdots \vee \phi_n \vee (\phi_n \rightarrow \chi_n)}, \tag{A.1}$$

preserves validity in Here and There.

Proof. Assume, by contradiction, that $\models \phi_1 \wedge \cdots \wedge \phi_n \rightarrow \chi_1 \vee \cdots \vee \chi_n$ but $\not\models (\phi_1 \rightarrow \chi_1) \vee \cdots \vee \phi_n \vee (\phi_n \rightarrow \chi_n)$. Thus there exists an HT model $\langle H, T\rangle$ such that:

- $\langle H, T\rangle \not\models \phi_i$, $1 \leq i \leq n$
- $\langle T, T\rangle \models \phi_i$, $1 \leq i \leq n$
- $\langle T, T\rangle \not\models \chi_i$, $1 \leq i \leq n$

From all the facts above and $\langle H, T\rangle \models \phi_1 \wedge \cdots \wedge \phi_n \rightarrow \chi_1 \vee \cdots \vee \chi_n$ we conclude $\langle T, T\rangle \models \chi_i$ for all i such that $1 \leq i \leq n$, reaching a contradiction ⊠

Lemma A.6. *Rules* (12)-(19) *from Table 4.1 preserve validity for all class of frames.*

Proof.

- **Rule** (12): let us assume that

$$\models \phi_1 \wedge \cdots \wedge \phi_m \rightarrow \chi_1 \vee \cdots \vee \chi_n \vee \varphi$$

but not

$$\not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \Box\varphi.$$

Since

$$\not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \Box\varphi$$

then, there exists a model $\mathcal{M} = \langle W, R^{\Box}, H, T\rangle$ and a world $x \in W$ such that

$$\mathcal{M}, x \not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \Box\varphi.$$

From

$$\mathcal{M}, x \not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \Box\varphi$$

we can derive the following conclusions:

(a) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, x \models \Box\phi_i$
(b) $\forall j\ 1 \leq i \leq n,\ \mathcal{M}, x \not\models \Diamond\chi_j$
(c) $\mathcal{M}, x \not\models \Box\varphi$

Moreover, (c) implies that there exists a world $y \in W$ such that $xR^{\Box}y$ and $\mathcal{M}, y \not\models \varphi$. However, from the fact $xR^{\Box}y$, (a) and (b), it follows that:

(d) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, y \models \phi_i$,
(e) $\forall j\ 1 \leq i \leq n,\ \mathcal{M}, y \not\models \chi_j$
(f) $\mathcal{M}, y \models \phi_1 \wedge \cdots \wedge \phi_m \rightarrow \chi_1 \vee \cdots \vee \chi_n \vee \varphi$

Finally, it is easy to see that (d), (e) and (f) imply the contradiction $\mathcal{M}, y \models \varphi$.

- **Rule** (13): let us assume that

$$\models \phi_1 \wedge \cdots \wedge \phi_m \rightarrow \chi_1 \vee \cdots \vee \chi_n \vee \neg\neg\varphi$$

but

$$\not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \neg\neg\Box\varphi,$$

which allows us to derive there exists a model $\mathcal{M} = \langle W, R^{\Box}, H, T\rangle$ and a Kripke world $x \in W$ such that

$$\mathcal{M}, x \not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n \vee \neg\neg\Box\varphi.$$

Hence, we conclude the following facts:

(a) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, x \models \Box\phi_i$,

(b) $\forall j\ 1 \leq j \leq n,\ \mathcal{M}, x \not\models \Diamond\chi_j$

(c) $\mathcal{M}, x \not\models \neg\neg\Box\varphi$, which is equivalent to $\mathcal{M}, x \not\models \Box\neg\neg\varphi$

From (c) we conclude that there exists $y \in W$ such that $xR^{\Box}y$ and $\mathcal{M}, y \not\models \neg\neg\varphi$. On the other hand, it also holds that:

(d) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, y \models \phi_i$

(e) $\forall j\ 1 \leq j \leq n,\ \mathcal{M}, y \not\models \chi_j$

(f) $\mathcal{M}, y \models \phi_1 \wedge \cdots \wedge \phi_m \wedge \rightarrow \chi_1 \vee \cdots \vee \chi_n \vee \neg\neg\varphi$

However, (d), (e) and (f) imply that $\mathcal{M}, y \models \neg\neg\varphi$, which is a contradiction.

- **Rule** (14): as for the previous cases, let us assume that

$$\models \phi_1 \wedge \cdots \wedge \phi_m \wedge \varphi \rightarrow \chi_1 \vee \cdots \vee \chi_n$$

but

$$\not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m\Diamond\varphi \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n,$$

so there exists a model $\mathcal{M} = \langle W, R^{\Box}, H, T\rangle$ and a Kripke world $x \in W$ such that

$$\mathcal{M}, x \not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m\Diamond\varphi \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n$$

and, as a consequence, it follows that:

(a) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, x \models \Box\phi_i$,

(b) $\mathcal{M}, x \models \Diamond\varphi$.

(c) $\forall j\ 1 \leq j \leq n,\ \mathcal{M}, x \not\models \Diamond\chi_j$

From (b) we conclude that, there exists a world $y \in W$ such that $xR^{\Box}y$ and $\mathcal{M}, y \models \varphi$. On the other hand, since $xR^{\Box}y$ we derive the following conclusions:

(d) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, y \models \phi_i$,

(e) $\forall j\ 1 \leq j \leq n,\ \mathcal{M}, y \not\models \chi_j$

(f) $\mathcal{M}, y \models \phi_1 \wedge \cdots \wedge \phi_m \wedge \varphi \rightarrow \chi_1 \vee \cdots \vee \chi_n$

Finally, from (d), (e) and (f) we conclude the contradiction $\mathcal{M}, y \not\models \varphi$.

- **Rule** (15): again, let us consider that

$$\models \phi_1 \wedge \cdots \wedge \phi_m \wedge \neg\neg\varphi \rightarrow \chi_1 \vee \cdots \vee \chi_n \tag{A.2}$$

but

$$\not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \wedge \neg\neg\Diamond\varphi \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n.$$

Therefore, there exists a model $\mathcal{M} = \langle W, R^{\Box}, H, T\rangle$ a world $x \in W$ such that

$$\mathcal{M}, x \not\models \Box\phi_1 \wedge \cdots \wedge \Box\phi_m \wedge \neg\neg\Diamond\varphi \rightarrow \Diamond\chi_1 \vee \cdots \vee \Diamond\chi_n.$$

Therefore, it follows that:

(a) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, x \models \Box\phi_i$,

(b) $\mathcal{M}, x \models \neg\neg\Diamond\varphi$, which is semantically equivalent to $\mathcal{M}, x \models \Diamond\neg\neg\varphi$,

(c) $\forall j\ 1 \leq j \leq n,\ \mathcal{M}, x \not\models \Diamond\chi_j$.

From (c), we conclude that there exists a world $y \in W$ such that $xR^{\Box}y$ and $\mathcal{M}, y \models \neg\neg\varphi$. On the other hand, $xR^{\Box}y$, (A.2), (a) and (c) imply that the following facts:

(d) $\forall i\ 1 \leq i \leq m,\ \mathcal{M}, y \models \phi_i$,

(e) $\forall j\ 1 \leq j \leq n,\ \mathcal{M}, y \not\models \chi_j$,

(f) $\mathcal{M}, y \models \phi_1 \wedge \cdots \wedge \phi_m \wedge \neg\neg\varphi \rightarrow \chi_1 \vee \cdots \vee \chi_n$

Finally the contradiction $\mathcal{M}, y \not\models \neg\neg\varphi$ follows from (d), (e) and (f).

Admissibility of rules (16)-(19) from Table 4.1 can be proven by following the same reasoning as for rules (12)-(15). ⊠

Lemma A.7. *The axioms* (20)-(23) *from Table 4.1 are valid for all class of frames.*

Proof. Let $\mathcal{M} = \langle W, R^{\Box}, R^{\bigcirc}, H, T\rangle$ be a modal HT model. We proceed by contradiction in all cases. In the case of Axiom (20) the proof goes as follows: while $\check{\mathcal{M}}, x \models \Box\varphi$ follows from $\mathcal{M}, x \not\models \Box\varphi \rightarrow \bot$, $\mathcal{M}, x \models \Diamond(\varphi \rightarrow \bot)$ together with the property of persistence imply $\check{\mathcal{M}}, x \models \Diamond(\varphi \rightarrow \bot)$. Since the latter derivation holds, it follows that

$$\exists y \in W \text{ s.t. } xR^{\Box}y \text{ and } \check{\mathcal{M}}, y \not\models \varphi.$$

Finally, from $xR^{\Box}y$ and $\check{\mathcal{M}}, x \models \Box\varphi$ we obtain the contradiction $\check{\mathcal{M}}, y \models \varphi$. In case of Axiom (21) were not valid, we could easily derive both $\check{\mathcal{M}}, x \not\models \Diamond\varphi$ and

$$\exists y \in W \text{ s. t. } xR^{\Box}y \text{ and } \check{\mathcal{M}}, x \models \varphi$$

Hence, from $\check{\mathcal{M}}, x \not\models \Diamond\varphi$ and $xR^{\Box}y$, we would find the contradiction $\check{\mathcal{M}}, y \not\models \varphi$.

Note that validity of axioms (22)-(23) from Table 4.1 are proved in a similar way but using the accessibility relation $R^{\bigcirc}$. ⊠

Lemma A.8. *Axioms* (24)-(27) *from Table 4.1 are valid with respect to the models $\mathcal{M} = \langle W, R^{\Box}, H, T\rangle$ where $R^{\Box}$ is reflexive and transitive.*

Proof. We proceed by assuming that these axioms are not valid, there exists a model $\mathcal{M} = \langle W, R^{\Box}, H, T\rangle$, and a Kripke point $x \in W$ where axioms Axioms (24)-(27) are not satisfied. We get a contradiction in all cases:

$$
\begin{array}{lcll}
\mathcal{M}, x \not\models (24) & \Rightarrow & \mathcal{M}, x \models \varphi \text{ and } \mathcal{M}, x \not\models \Diamond\varphi & \\
& \Rightarrow & \mathcal{M}, x \models \varphi \text{ and } \mathcal{M}, x \not\models \varphi & (R^{\Box} \text{ is reflexive}) \\
\\
\mathcal{M}, x \not\models (25) & \Rightarrow & \mathcal{M}, x \models \Box\varphi \text{ and } \mathcal{M}, x \not\models \varphi & \\
& \Rightarrow & \mathcal{M}, x \models \varphi \text{ and } \mathcal{M}, x \not\models \varphi & (R^{\Box} \text{ is reflexive}) \\
\\
\mathcal{M}, x \not\models (26) & \Rightarrow & \mathcal{M}, x \models \Box\varphi \text{ and } \mathcal{M}, x \not\models \Box\Box\varphi & \\
& \Rightarrow & \begin{cases} \mathcal{M}, x \models \Box\varphi \text{ and } \exists\ y,\ z \in W \text{ s.t.} \\ xR^{\Box}y,\ yR^{\Box}z \text{ and } \mathcal{M}, z \not\models \varphi \end{cases} & \\
& \Rightarrow & \mathcal{M}, z \models \varphi \text{ and } \mathcal{M}, z \not\models \varphi & (R \text{ is transitive}) \\
\\
\mathcal{M}, x \not\models (27) & \Rightarrow & \mathcal{M}, x \models \Diamond\Diamond\varphi \text{ and } \mathcal{M}, x \not\models \Diamond\varphi & \\
& \Rightarrow & \begin{cases} \mathcal{M}, x \not\models \Diamond\varphi \text{ and } \exists\ y,\ z \in W \text{ s.t.} \\ xR^{\Box}y,\ yR^{\Box}z \text{ and } \mathcal{M}, z \models \varphi \end{cases} & \\
& \Rightarrow & \mathcal{M}, z \not\models \varphi \text{ and } \mathcal{M}, z \models \varphi & (R^{\Box} \text{ is transitive})
\end{array}
$$

⊠

Lemma A.9.
The axioms (28)-(29) *are valid with respect to the class of models* $\mathcal{M} = \langle W, R^{\Box}, R^{\bigcirc}, H, T\rangle$ *satisfying the property*

$$\textit{If } x \left(R^{\bigcirc}\right)^{*} y \textit{ then } xR^{\Box}y. \tag{A.3}$$

Proof. We proceed by contradiction.

$$
\begin{array}{lcll}
\mathcal{M}, x \not\models (28) & \Rightarrow & \mathcal{M}, x \not\models \bigcirc\varphi & \\
& \Rightarrow & \exists y \in W \text{ s. t. } xR^{\bigcirc}y \text{ and } \mathcal{M}, y \not\models \varphi & \\
& \Rightarrow & xR^{\Box}y \text{ and } \mathcal{M}, y \not\models \varphi & \text{(A.3)} \\
& \Rightarrow & \mathcal{M}, y \models \varphi \text{ and } \mathcal{M}, y \not\models \varphi & (\mathcal{M}, x \models \Box\varphi \text{ and } xR^{\Box}y) \\
\\
\mathcal{M}, x \not\models (28) & \Rightarrow & \mathcal{M}, x \models \hat{\bigcirc}\varphi & \\
& \Rightarrow & \exists y \in W \text{ s. t. } xR^{\bigcirc}y \text{ and } \mathcal{M}, y \models \varphi & \\
& \Rightarrow & xR^{\Box}y \text{ and } \mathcal{M}, y \models \varphi & \text{(A.3)} \\
& \Rightarrow & \mathcal{M}, y \models \varphi \text{ and } \mathcal{M}, y \not\models \varphi & (\mathcal{M}, x \not\models \Diamond\varphi \text{ and } xR^{\Box}y)
\end{array}
$$

Hence, we reach a contradiction in all cases. ⊠

Lemma A.10. *Rules of inference* (30)-(31) *from Table 4.1 preserve validity with respect to the class of models* $\mathcal{M} = \langle W, R^{\Box}, R^{\bigcirc}, H, T\rangle$ *that satisfy the property*

$$\textit{If } xR^{\Box}y \textit{ then } x \left(R^{\bigcirc}\right)^{*} y \tag{A.4}$$

Proof.

- **Rule** (30): assume that $\models \varphi \rightarrow \Box\varphi$ but $\not\models \varphi \rightarrow \bigcirc\varphi$. From the latter assumption we conclude that $\mathcal{M}, x \models \varphi$ but $\mathcal{M}, x \not\models \Box\varphi$ for some $x \in W$. Thus, we derive

$$\exists y \in W. \text{ s.t. } xR^{\Box}y \text{ and } \mathcal{M}, x \not\models \varphi.$$

Now, applying (A.4) to $xR^{\Box}y$, we get

$$\exists n \in \mathbb{N} \text{ and } y \in W. \text{ s.t. } x \left(R^{\bigcirc}\right)^n y \text{ and } \mathcal{M}, y \not\models \varphi.$$

Now, in order to get a contradiction, we claim that $\mathcal{M}, y \models \varphi$. We prove this by induction on n

- **Base case (n=0):** If $n = 0$ then $x = y$ and, since $\mathcal{M}, x \models \varphi$ then $\mathcal{M}, y \models \varphi$.
- **Inductive step:** Assume that $x \left(R^{\bigcirc}\right)^n y$ and

  $$\forall z \in W. \text{ if } x \left(R^{\bigcirc}\right)^{n-1} z \text{ then } \mathcal{M}, z \models \varphi.$$

  Since $x \left(R^{\bigcirc}\right)^n y$ can be rewritten as $x \left(R^{\bigcirc}\right)^{n-1} t$ and $tR^{\bigcirc}y$ (with $t \in W$), then by induction hypothesis we know that $\mathcal{M}, t \models \varphi$. Finally, from $\mathcal{M}, t \models \varphi$ and $\models \varphi \rightarrow \bigcirc\varphi$ we conclude that $\mathcal{M}, y \models \bigcirc\varphi$, which together with the fact $tR^{\bigcirc}y$ implies that $\mathcal{M}, y \models \varphi$.

Therefore $\mathcal{M}, y \models \varphi$ contradicts the previous fact $\mathcal{M}, y \not\models \varphi$.

- **Rule** (31): we proceed by following a similar reasoning task. Let us assume that $\models \hat{\bigcirc}\varphi \rightarrow \varphi$ but $\not\models \Diamond\varphi \rightarrow \varphi$. Hence, from $\not\models \Diamond\varphi \rightarrow \varphi$ we can derive

  $$\exists x \in W \text{ s. t. } \mathcal{M}, x \models \Diamond\varphi \text{ but } \mathcal{M}, x \not\models \varphi$$

  and, since $\mathcal{M}, x \models \Diamond\varphi$, then

  $$\exists y \in W \text{ s. t. } xR^{\Box}y \text{ and } \mathcal{M}, y \models \varphi.$$

  Now, by (A.4), we obtain the following expression

  $$\exists n \in \mathbb{N}, y \in W \text{ s. t. } x \left(R^{\bigcirc}\right)^n y \text{ and } \mathcal{M}, y \models \varphi.$$

  ITo get a contradiction, we claim that $\mathcal{M}, y \not\models \varphi$ and we will use induction on n to prove such claim:

  - **Base case (n=0)**: if $n = 0$ then $y = x$ and, since $\mathcal{M}, x \not\models \varphi$, we conclude $\mathcal{M}, y \not\models \varphi$.
  - **Inductive step:** Assume that $x \left(R^{\bigcirc}\right)^n y$ and

    $$\forall z \in W. \text{ if } x \left(R^{\bigcirc}\right)^{n-1} z \text{ then } \mathcal{M}, z \not\models \varphi.$$

    Since $x \left(R^{\bigcirc}\right)^n y$ can be rewritten as $x \left(R^{\bigcirc}\right)^{n-1} t$ and $tR^{\bigcirc}y$ (with $t \in W$), then by induction hypothesis we know that $\mathcal{M}, t \not\models \varphi$. Finally, from $\mathcal{M}, t \not\models \varphi$ and $\models \hat{\bigcirc}\varphi \rightarrow \varphi$ we conclude that $\mathcal{M}, y \not\models \hat{\bigcirc}\varphi$, which together with the fact $tR^{\bigcirc}y$ implies that $\mathcal{M}, y \not\models \varphi$.

  Since the claim is proved we can assure that $\mathcal{M}, y \not\models \varphi$, reaching a contradiction.

⊠

Lemma A.11. *Axiom* (32) *is valid with respect to the class of models* $\mathcal{M} = \langle W, R^{\bigcirc}, H, T\rangle$ *such that* $R^{\bigcirc}$ *is linear, that is, for every Kripke world* x *in* W *there exists one and only one successor denoted by* $succ(x)$.

Proof. Assume that Axiom (32) is not satisfied at $x \in W$, thus we have

$$\exists x \in W \text{ s.t. } \mathcal{M}, x \not\models \bigcirc\varphi \leftrightarrow \hat{\bigcirc}\varphi.$$

Therefore we must check four different cases:

1. $\mathcal{M}, x \models \bigcirc\varphi$ but $\mathcal{M}, \not\models \hat{\bigcirc}\varphi$: we derive the contradiction $\mathcal{M}, succ(x) \models \varphi$ and $\mathcal{M}, succ(x) \not\models \varphi$. From the former and the latter respectively.
2. $\mathcal{M}, x \not\models \bigcirc\varphi$ but $\mathcal{M}, \models \hat{\bigcirc}\varphi$: we obtain the contradiction $\mathcal{M}, succ(x) \not\models \varphi$ and $\mathcal{M}, succ(x) \models \varphi$ from the former and the latter respectively.
3. $\check{\mathcal{M}}, x \models \bigcirc\varphi$ but $\check{\mathcal{M}}, \not\models \hat{\bigcirc}\varphi$: as happens in the previous cases we conclude $\check{\mathcal{M}}, succ(x) \models \varphi$ and $\check{\mathcal{M}}, succ(x) \not\models \varphi$ from both initial assumptions.
4. $\check{\mathcal{M}}, x \not\models \bigcirc\varphi$ but $\check{\mathcal{M}}, \models \hat{\bigcirc}\varphi$: we conclude $\check{\mathcal{M}}, succ(x) \not\models \varphi$ and $\check{\mathcal{M}}, succ(x) \models \varphi$ from the initial assumptions.

Hence, since we have reached a contradiction in all cases, we proved that Axiom (32) is valid with respect to the class of linear frames. ⊠

A.4 Properties of the canonical model

Proposition A.1. *Given two systems* x, y *belonging to the canonical model* $\mathcal{M}_c$. *If* $xR_c^{\bigcirc}y$ *then* $\check{x}R_c^{\bigcirc}\check{y}$.

Proof. Since $xR_c^{\bigcirc}y$ we get, by Lemma 4.7 that both $\bigcirc\Gamma_T^x \subseteq \Gamma_T^y$ and $\hat{\bigcirc}\Delta_T^x \subseteq \Delta_T^y$, but it is equivalent to say that $\bigcirc\Gamma_H^{\check{x}} \subseteq \Gamma_H^{\check{y}}$ and $\hat{\bigcirc}\Delta_H^{\check{x}} \subseteq \Delta_H^{\check{y}}$. Hence, by definition, $\check{x}R_c^{\bigcirc}\check{y}$.

⊠

Proposition A.2. *Given two systems* x, y *belonging to the canonical model* $\mathcal{M}_c$. *If* $xR_c^{\Box}y$ *then* $\check{x}R_c^{\Box}\check{y}$.

Proof. The proof can be done by following the same reasoning as in Proposition A.1. ⊠

Appendix B

The STeLP system

STeLP[1] is an on-line application designed for computing Temporal Equilibrium Models of *splittable temporal logic programs* [3] but adding new features like the use of variables and the possibility of declaring arbitrary LTL formulas in the body of a constraints.

B.1 Syntax

STeLP allows defining the following three different types of rules:

- `initial_0`: these are rules of the form

  $$\neg p_1 \wedge \cdots \neg p_m \wedge p_{m+1} \wedge \cdots \wedge p_n \rightarrow a_1 \vee \cdots \vee a_p$$

  such that

  - the use of negation is not allowed in the consequent.
  - the use of operator $\bigcirc$ is not allowed neither in the antecedent nor in the consequent.

  Initial_0 rules are represented in STeLP as in ASP:

  ```
  a1 v ... v ap :- neg p1, ..., neg pm, not pm+1, ... not pn
  ```

- `initial_1`: these are rules of the form

  $$\begin{aligned} &\neg p_1 \wedge \cdots \neg p_m \wedge \neg \bigcirc p_{m+1} \wedge \cdots \wedge \neg \bigcirc p_n \\ &\quad \wedge p_{n+1} \wedge \cdots \wedge p_t \wedge \bigcirc p_{t+1} \wedge \cdots \wedge \bigcirc p_s \quad \rightarrow \quad \bigcirc a_1 \vee \cdots \vee \bigcirc a_p \end{aligned}$$

  such that

  - the use of negation is not allowed in the consequent.
  - $\bigcirc$ operator modifies every atom in the consequent.

[1] Available at `http://kr.irlab.org/stelp_online/`

STeLP representation would corresponds to the following rule:

```
o a1 v ... v o ap :- neg p1, ..., neg pm, o not pm+1, ...,
                     o not pn,  pn+1, ..., pt, opt+1... o ps.
```

- dynamic: this rules are either initial_1 or initial_0 but they are under the scope of operator $\Box$, that is, they are of the form

$$\Box\,(\neg p_1 \wedge \cdots \neg p_m \wedge p_{m+1} \wedge \cdots \wedge p_n \quad \rightarrow \quad a_1 \vee \cdots \vee a_p)$$

  where, the use of $\bigcirc$ is allowed but if an atom in the antecedent is affected by it then this operator has to modify every atom in the consequent. As in the rest of the cases, the use of negation is not allowed in the consequent.

  Representation of dynamic rules is made by replacing the classical "if" from logic programming by the symbol "::-". For instance the rule above would be equivalent to the next one, written in STeLP format:

```
a1 v ... v ap :- neg p1, ..., neg pm, not pm+1, ...,
                 not pn,  pn+1, ..., pt, pt+1... ps.
```

B.2 Implementation

In a first step, STeLP, whose modules are shown in Figure B.1, behaves as follows: it firstly parses the input program, detecting static and domain predicates, checking safety of all rules, and separating the static rules. It also appends static predicates to rule bodies for all variables with global domain. The set of static rules is fed to DLV to generate some model (among possible) that will provide the extension for all static predicates. The number of static models that STeLP will consider is given as a command line argument. Each static model will generate a different ground program and a different automaton. Once a static model is fixed, STeLP grounds the non-static rules input program. Each ground instance is evaluated and, if the body of the ground rule becomes false, the rule is deleted. Otherwise all static predicates of the rule are deleted from its body (see Algorithm 3).

The next step computes the loop formulas for the ground STLP we have obtained by constructing a dependency graph and obtaining its strongly connected components. The STLP plus its loop formulas are then used as input for the LTL solver SPOT, which returns a deterministic Büchi automaton. Finally, STeLP parses and, if actions and fluents are defined, simplifies the automaton as described before. The tool Graphviz[2] is used for generating a graphical representation.

[2] http://www.graphviz.org/

Algorithm 3: Temporal_Grounding(Π)

Require: A first-order splittable logic program Π
Ensure: list of possible ground rules Π_g

```
Π_g := ∅
complete_with_domain_predicates(Π)
Π_1 := select_static_rules(Π)
Π_2 := Π \ Π_1
M := solve(Π_1)
for all m ∈ M do
  Gr := ∅
  for all r ∈ Π_2 do
    L:=compute_valid_substitutions(Π_2, m)
    for all θ ∈ L do
      r_g:= apply_substitution(r, Θ)
      r_g:= remove_static_predicates(r_g)
      Gr := Gr ∪ {r_g}
    end for
  end for
  Π_g := Π_g ∪ Gr
end for
return Π_g
```

B.3 Use of variables, actions and fluents

`STeLP` allows using variables in the representations but, to guarantee the program to be domain independent, they must appear in a positive *static* predicate in the body of the rule. User must specify which predicates are static by using the reserved word `static` following by a list of items of the form `name/arity`, where name and arity correspond to the name and arity of the static predicate respectively. Another way of declaring static predicates is by means of the reserved word `domain`, which is followed by a list of predicates (with its variables). Every predicate in the list is implicitly declared as static and, moreover, whenever a variable occurring in the program also occurs in a domain predicate then such domain predicate is automatically added to the body. We recommend to check section B.4.

Apart from the use of variables, this tool provides two reserved words, `fluent` and `action`. In order to get a more clear representation, every atom, declared as fluent, is moved to the state and every atom declared as an action is left in the transitions. Again, in section B.4 we provide an example of specification of fluents and actions.

B.4 Using `STeLP` for Model Checking

`STeLP` can be used for model checking since arbitrary LTL formulas can be encoded in the body of a rule.

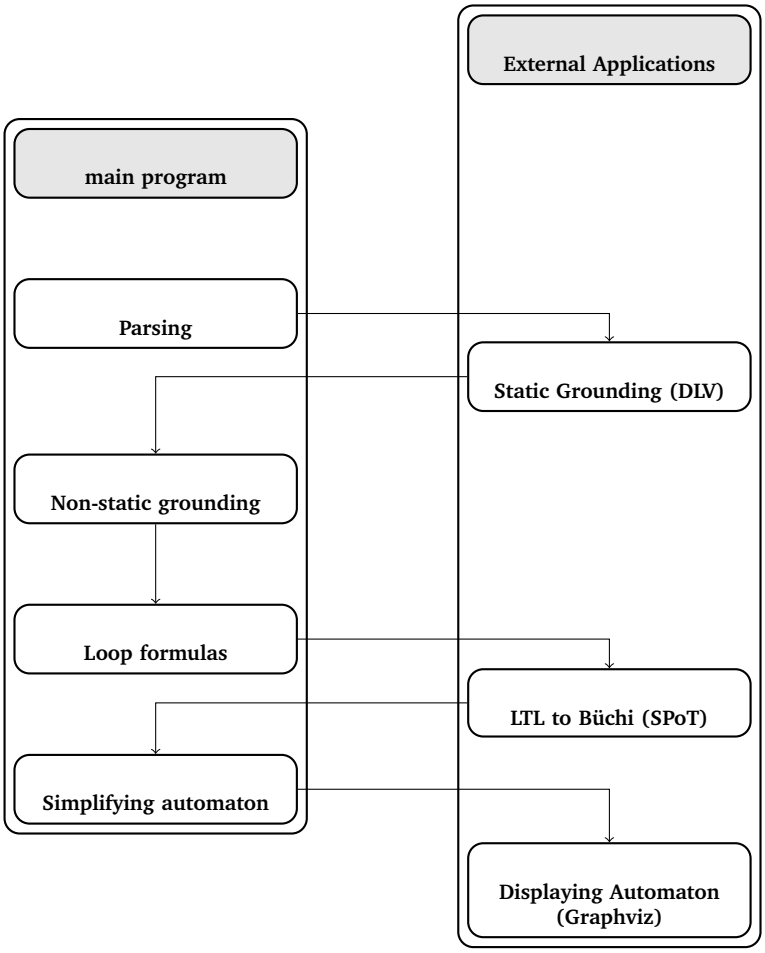


Figure B.1: Structure of STeLP system.

Example B.1 (from [61])**.** *Two concurrent processes share a resource (such as a file on a disk or a database entry), it may be necessary to ensure that they do not have access to it at the same time. Several processes simultaneously editing the same file would not be desirable.*

To avoid this problem we stablish several critical sections *of each process' code and design a protocol that satisfies the following conditions :*

- *Only one process can be in its critical section at a time.*
- *Critical section should include all the access to the shared resource (though it should be as small as possible so that no unnecessary exclusion takes place)*

However, this protocol is not verified yes. It should satisfy some the following desired properties

- *Safety: only one process is in its critical section at a time.*

- *Liveness: whenever any process requests to enter its critical section, it will eventually be permitted to do so.*

⊠

We use STeLP to represent to represent this problem, whose encoding is shown next.

```
domain proc(X), proc(Y).
domain nextip(I,J).
domain inst(A), inst(B).
fluent ip/2.
action sch/1.
proc(1).      proc(2).
nextip(n,t). nextip(t,c). nextip(c,n).
inst(n). inst(t). inst(c).
% Effect axioms
o ip(X,J) ::- sch(X), ip(X,I).
% State constraint\n"
::- ip(X,c), ip(Y,c), X'!='Y.
% Unique value
::- ip(X,A), ip(X,B), A'!='B.
% Inertia
o ip(X,A) ::- ip(X,A), not o other(X,A).
other(X,A) ::- ip(X,B), A'!='B.
% Action generation
aux(X) ::- not sch(X).
sch(X) ::- not aux(X).
::- sch(X), sch(Y), X'!='Y.
somesch ::- sch(X).
::- not somesch.
ip(X,n).
```

In this representation, predicates `proc` and `sch` represent the processes and the action schedule respectively. Predicate stands for "interrupted process" whose situation inside the critical section is represented by the constants 'n' (not in the critical section), 't' (waiting to enter in the critical section) and 'c' (inside the critical section). The fixed sequences of states that a process follows inside the interruption, that is, from being in the non critical section to being inside the critical section is defined by the extensional predicate `nextip`. We must add constraints in order to forbid possible configurations like those where both process are at the critical section at the same time. Temporal equilibrium models are shown in Figure B.2.

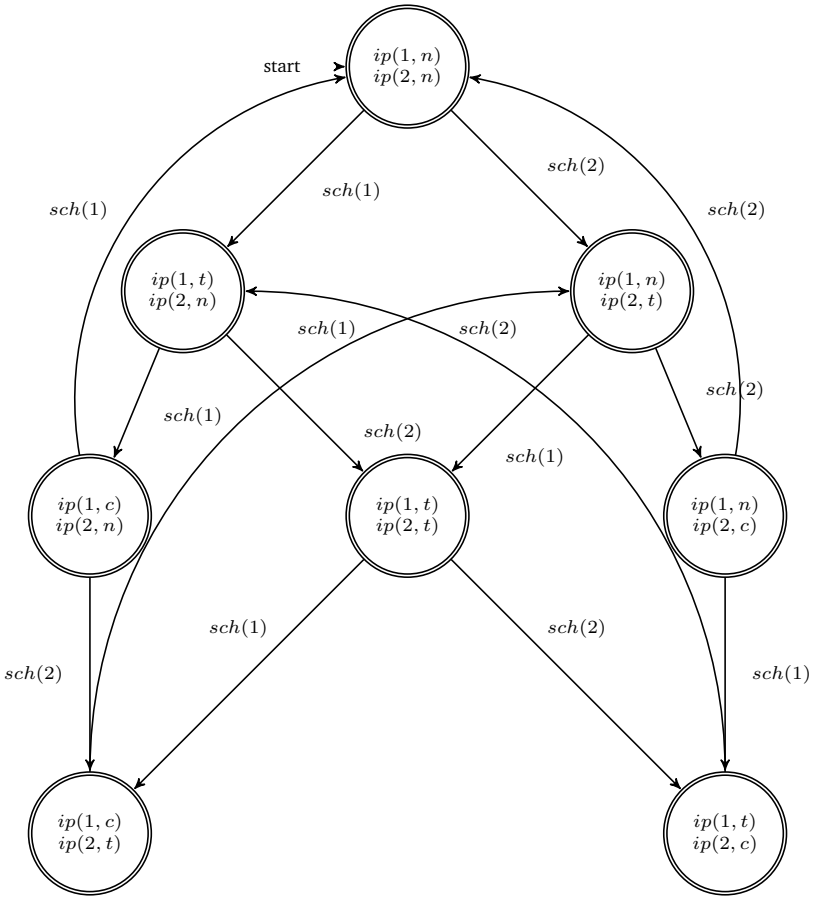


Figure B.2: Büchi automaton that represents the protocol specified in Example B.1

As an example of how to check temporal properties by using constraints, let us check whether the system represented is safe. It would mean checking that program satisfies the formula

$$\varphi = \Box\neg(ip(1,c) \wedge ip(2,c)).$$

Assume that we add $\neg\varphi$ to our representation. It would mean that every THT $\langle \mathbf{H}, \mathbf{T} \rangle$ should satisfy $\neg\varphi$ or, what is the same, φ is not satisfied in LTL. In this way, the automaton obtained, shown in Figure B.3 does not accept any temporal equilibrium models, and this fact leads to determine that the system is safe.

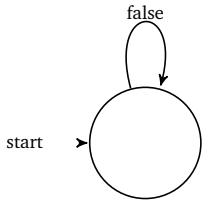


Figure B.3: Büchi automaton resulting from adding the formula $\neg\varphi$.

The property of liveness, which corresponds to the formula

$$\Box\left(\neg p(1,t) \wedge \neg\Box\neg ip(1,c)\right) \wedge \Box\left(\neg p(2,t) \wedge \neg\Box\neg ip(2,c)\right),$$

can be checked in the same way.

Appendix C

The ABSTEM system

The tool ABSTEM[1] is a tool designed to compute the set of temporal equilibrium models as well as checking several temporal properties on arbitrary theories [18, 20]. ABSTEM has dependencies with both SPoT[2], a library to operate with ω-automata, and Graphviz, an open source graph visualization sofware. Once these dependencies are met, ABSTEM can be easily compiled using the make command:

```
cd path-to-abstem.
make && make install
```

C.1 Input Syntax

Arbitrary theories must be written following SPoTsyntax, which is shown in Table C.1. for instance, the formula

$$\neg\left((p \wedge q)\,\mathcal{U} r\right) \tag{C.1}$$

would correspond to the formula

```
! ((p & q) U r)
```

written in ABSTEM syntax. Input formulas can written directly as an input parameter (by means of `-c` option) or in a file whose path must be provided by means of the command-line option `-f`.

[1] available at `http://kr.irlab.org/?q=abstem`.

[2] A modified version of SPoTwhich avoids some simplifications like, for instance, $\neg\neg\varphi \Leftrightarrow \varphi$, that are valid in LTL but not in THT.

Formula	**Syntax**
$\bot$	false or 0
$\top$	true or 1
$\neg\varphi$	! φ or $\sim \varphi$
$\varphi \wedge \varphi'$	φ & φ'
$\varphi \vee \varphi'$	$\varphi \mid \varphi'$
$\varphi \rightarrow \varphi'$	φ -> φ'
$\varphi \leftrightarrow \varphi'$	φ <-> φ'
$\Box\varphi$	[ ]φ or G φ
$\Diamond\varphi$	<>φ or F φ
$\bigcirc\varphi$	X φ
$\varphi \,\mathcal{U}\, \varphi'$	φ U φ'
$\varphi \,\mathcal{R}\, \varphi'$	φ R φ'

Table C.1: ABSTEM input syntax.

C.2 Computing Temporal Equilibrium Models

Option `-t` enables the option of computing temporal equilibrium models. The algorithm implemented [18] computes several intermediate automata that can be displayed by using the following command-line options:

i) `--SForm:` generates a file named `total.png` which contains the Büchi automaton that accepts the LTL models of an input formula.

ii) `--SNonTot:` produces a file named `non_total.png` that shows the Büchi automaton that accepts the non-total THT models of the input formula.

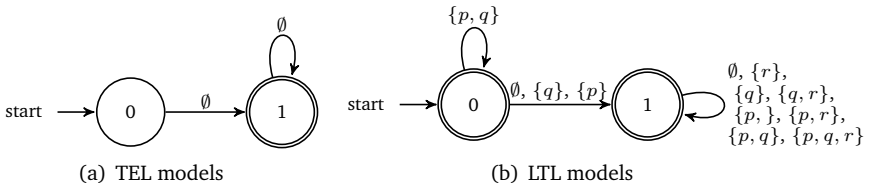
iii) `--SFiltered:` create a file named `filtered.png` that contains the Büchi automaton resulting from filtering ii)

iv) `--SComp:` displays the complementary automaton of iii) in a file named `complementary.png`

Finally, temporal equilibrium models are shown in a file designated as `tem.png`. Due to the state explosion when complementing an automaton, the final automaton may contain so many states which difficult its understanding. However, by using option `-m` every automaton is simplified before being written into a file. As an example, let us consider the formula (C.1), by executing the command-line option

```
abstem -t -m -c '!  ((p & q) U r)'
```

we obtain the automaton of the Figure C.1



(a) TEL models (b) LTL models

Figure C.1: THT and LTL models of $\neg((p \wedge q)\,\mathcal{U} r)$

C.3 Strong Equivalence

ABSTEM allows checking three different kinds of equivalence between two temporal formulas. Those kinds can be enabled by means of the following command-line options:

- `-l`: enables to check LTL equivalence.
- `-w`: enables to check Weak equivalence.
- `-s`: enables to check Strong equivalence.

For instance, we can check that formulas $\Diamond p$ and $\neg\Box\neg p$ are LTL equivalent but are not strongly equivalent. In case of strong equivalence, ABSTEM fixes the context $\gamma = \Diamond p \rightarrow \Box(p \vee \neg p)$ that produces a different behaviour, which is displayed in the automaton of Figure C.2

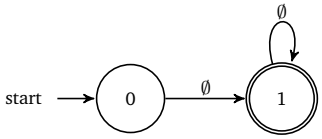


Figure C.2: TEM models of $\Diamond p \wedge \gamma$ but not of $\neg\Box\neg p \wedge \gamma$.

Apéndice D

Resumen

Hans Kamp comenzó su famosa tesis doctoral [63] en lógica temporal con la siguiente frase:

> "Grecia es un reino, es cierto hoy (10 de mayo de 1968); pero era falso antes y ¿quién sabe si será verdad mañana?"

Mediante esta frase, Kamp pretendía enfatizar que la certeza de una frase podría depender del momento en el que se dice. El estaba probablemente sospechando que el rey Constantino II de Grecia, en aquel momento reinando desde el exilio en 1968, podría ser cesado, como realmente ocurrió en junio de 1973, siendo el fin de la monarquía griega. Si nos preguntasen sobre el sistema político griego en cualquier instante de tiempo entre 1968 y 1973, hubiéramos respondido monarquía, porque nosotros no tenemos mayores evidencias de lo contrario. Esto es un ejemplo de conclusión por defecto, que puede cambiar a la vista de nuevas evidencias. Consideremos ahora, por ejemplo, el caso del rey Baudouin de Bélgica, quien oficialmente reinó desde el 17 de julio de 1951 al 31 de julio de 1993. Sin más información, podemos asumir que él estuvo en el cargo en cualquier fecha dentro de dicho periodo. Sin embargo, es conocido que el 4 de abril de 1990 él abdicó durante un día, de acuerdo con el gobierno belga, para no tener que firmar la ley del aborto. Después de añadir este hecho, nosotros deberíamos poder retractarnos de nuestra conclusión previa y, a su vez, derivar que el rey no reinó durante dicho día.

El razonamiento del sentido común en el caso temporal está lleno de situaciones que requieren asumir conclusiones por defecto, puesto que raramente contamos con toda la información disponible. Lamentablemente, la mayoría de lógicas modales temporales no permiten modelar este tipo de razonamiento por defecto debido a que, típicamente, se definen por medio de relaciones de inferencia monótonas (como pasa también en lógica clásica). Formalmente, esto significa que si una formula φ se deriva de una teoría Γ, entonces φ seguirá siendo derivable de una teoría de la forma $\Gamma \cup \Delta$. En otras palabras, una conclusión φ que es derivable de un conjunto de fórmulas Γ *no puede ser retractada* al añadir una nueva evidencia Δ.

La lógica computacional ha sido aplicada al razonamiento temporal bajo dos perspectivas diferentes: teoría de la computación (TC) y representación

del conocimiento (RC). Debido a que los objetivos de estas dos áreas son bastante diferentes, los resultados obtenidos en ambos casos han tomado caminos distintos. En TC, el razonamiento temporal está más orientado al estudio de propiedades de algoritmos y sus problemas asociados tales como complejidad, computabilidad, verificación formal, conexión con otros formalismos como teoría de autómatas, álgebra o lenguajes formales. En todos estos casos, el uso de diferentes tipos de *lógicas modales* ha demostrado ser extremadamente útil, ya que proporcionan construcciones específicas para hablar sobre tiempo, a la vez que restringen la expresividad de la lógica de predicados a una subclase decidible. Por el contrario, en RC, la investigación ha sido tradicionalmente enfocada hacia la resolución de problemas de razonamiento donde las reglas por defecto y el *razonamiento no monótono* (RNM) han jugado un papel crucial. En particular, la resolución de este tipo de problemas ha sido el principal tema de investigación de la subárea de RC denominada *razonamiento sobre acciones y cambio* (RAC), donde los más famosos lenguajes formales no son más que dialectos de la lógica de primer orden, debido a que se pretende una rica capacidad expresiva.

Mientras que la no monotonicidad apenas ha sido considerada dentro del discurso en TC, su combinación con lógicas modales en RC no es extraño. Por ejemplo, existe una familia de *lógicas modales no monótonas* (ver por ejemplo [87]) que se obtienen mediante la imposición de condiciones de punto fijo en la relación de inferencia de las lógicas modales estándar. No obstante, este tipo de lógicas ha tenido un uso de carácter epistémico, representando conceptos como conocimiento, creencia u obligación, manejando únicamente los operadores modales de necesidad y posibilidad. Otras combinaciones de no monotonicidad y lógicas modales especificamente diseñadas para razonamiento temporal es mucho más infrecuente en la literatura, las únicas excepciones son típicamente los lenguajes de acciones modales con semánticas no monótonas definidas bajo diversas restricciones sintácticas.

Hasta donde sabemos, el único caso de una lógica temporal no monótona que cubra la sintaxis de alguna de las aproximaciones modales tradicionales sin requerir el uso de más construcciones es *Temporal Equilibrium Logic* (TEL), lógica definida por Cabalar y Pérez en [23]. TEL comparte la sintaxis de *Linear-time Temporal Logic* (LTL) (formalismo propuesto por Arthur Prior [103] y posteriormente extendido por Kamp [63]), que es una de las lógicas más simples, utilizadas y mejor conocidas en TC. La principal diferencia con TEL está en la interpretación semántica de la implicación y de la negación, que está más próxima a la lógica intuicionista. Estas dos propiedades están heredadas del hecho de que TEL es una extensión temporal de *Equilibrium Logic* [98], un formalismo no monótono que ofrece una generalización de la semántica de *modelos estables* [48] de programación lógica para el caso de fórmulas proposicionales arbitrarias. Dicha semántica es una característica importante por diversas razones. Primero, los modelos estables constituyen la base del exitoso paradigma de razonamiento no monótono denominado *Answer Set Programming* (ASP), avalado por el desarrollo de herramientas de resolución muy eficientes y, además, aplicadas en dominios muy heterogéneos. Segundo, mientras que en la definición original de modelos estables, la semántica de Equilibrium Logic no depende de transformaciones sintácticas sino que consiste, únicamente,

en un criterio de minimización de modelos para una lógica intermedia (la lógica Here-and-There [57]). Esta definición puramente lógica proporciona una forma más fácil y homogénea de extender el formalismo, utilizando las técnicas estándar de otras lógicas híbridas. Así, por ejemplo, la extensión temporal en TEL puede verse como la forma "obvia" de introducir operadores de LTL en Equilibrium Logic.

El primer par de trabajos sobre TEL exploraron su uso en la traducción de lenguajes de acciones [23] o su aplicación potencial para el chequeo de la propiedad de *(strong) equivalence* entre representaciones alternativas de algún dominio de acciones [2]. Sin embargo, aparte de la propia definición de TEL muy pocos resultados, relativos a sus propiedades fundamentales, eran conocidos [17] y nada en absoluto sobre qué métodos de cálculo se podrían utilizar en casos de aplicación. Esta situación contrastaba con el vasto conocimento disponible sobre LTL, tanto relativo a sus propiedades formales como a sus métodos de cómputo e implementaciones prácticas.

D.1 Metodología

La metodología utilizada en esta tesis se corresponde con la estándar en el campo de la investigación en ciencias de la computación, una secuencia cíclica incluyendo: revisión del estado del arte, definición del problema, establecimiento de hipótesis, derivación de una prueba formal o refutación, desarrollo y chequeo de prototipos (en caso de temas de implementación) y finalmente evaluación y publicación de los resultados. Muchos resultados obtenidos se acompañan por prototipos software (especialmente en los casos de `STeLP` y `ABSTEM`), para los que hemos empleado un ciclo de vida en espiral, pasando por diferentes ciclos de análisis, diseño, implementación y evaluación. Como resultado, esta tesis ha generado las siguientes publicaciones:

- P. Cabalar and M. Diéguez. *STeLP - A Tool for Temporal Answer Set Programming*, in Proc. of the 11$^{\text{th}}$ International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'11), pages 370–375, Vancouver, Canada, 2011.

- P. Cabalar and M. Diéguez. *Strong equivalence of non-monotonic temporal theories*, in Proceedings of the 14$^{\text{th}}$ International Conference on Principles of Knowledge Representation and Reasoning (KR'14), Vienna, Austria, 2014.

- F. Aguado, P. Cabalar, M. Diéguez, G. Pérez, C. Vidal, *Temporal equilibrium logic: a survey*. Journal of Applied Non-Classical Logics 23(1-2): 2-24 (2013).

- F. Aguado, P. Cabalar, M. Diéguez, G. Pérez and C. Vidal, *Paving the Way for Temporal Grounding*, in Proc. of the 28$^{\text{th}}$ International Conference on Logic Programming (ICLP'12), Budapest, Hungary, 2012.

D.2 Resultados obtenidos

En esta tesis hemos estudiado diferentes aspectos de Temporal Equilibrium Logic, una novedosa combinación de lógica modal temporal y un formalismo no monótono. A grandes rasgos, esta tesis recoge un conjunto de resultados que constituye un gran logro en lo relativo al conocimiento sobre TEL. Las principales contribuciones de esta tesis son las siguientes:

i Hemos probado el teorema de *strong equivalence* para el caso de TEL, lo que justifica el uso de THT como base monótona.

ii Hemos demostrado que la traducción sintáctica propuesta por Kamp para codificar LTL en lógica de primer order es correcta para codificar TEL en Quantified Equilibrium Logic [101].

iii Hemos propuesto una codificación de TEL en el formalismo *Infinitary Equilibrium Logic* [55] y, por lo tanto es posible aplicar la idea de *reducto* definido para este formalismo al caso de teorías TEL arbitrarias.

iv Hemos realizado la axiomatización de una parte relevante de la base monótona de Temporal Here and There, la base monótona de TEL. En particular hemos proporcionado un conjunto de axiomas para los operadores "necesario", "posible" y "siguiente" que propiamente fija su comportamiento semántico (exceptuando la linealidad del "siguiente" que es dejada como trabajo futuro).

v Se ha desarrollado una herramienta llamada `STeLP` que acepta como entrada teorías temporales que pertenecen a una subclase sintáctica de TEL, denominada *splittable*, y computa sus modelos temporales de equilibrio en la forma de un autómata de Büchi. Esta subclase de TEL se caracteriza por la propiedad de que una referencia al pasado nunca puede depender de una referencia al futuro, lo que es suficiente para cubrir la mayoría de los escenarios que se pueden representar en Answer Set Programming.

vi Hemos definido la versión primer orden de TEL así como un algoritmo de *grounding* que permite la eliminación de variables en el caso de programas lógicos *splittable*.

vii Hemos desarrollado otra herramienta, llamada `ABSTEM`, que se centra en el cálculo de modelos temporales de equilibrio de cualquier teoría arbitraria y, además, permite comprobar diversos tipos de equivalencia entre teorías temporales. Además, en caso de que las teorías no sean equivalentes esta herramienta produce una justificación en términos de contramodelos.

viii Finalmente, hemos comparado TEL con otras aproximaciones no monótonas para la representación de escenarios temporales. Utilizando la traducción en *Infinitary Equilibrium Logic* hemos sido capaces de probar que la semántica de TEMPLOG [11], una aproximación de programación lógica temporal, sin negación por defecto, coincide con el resultado

de TEL para dicha subclase sintáctica. Nosotros hemos extendido esta comparación al formalismo *Temporal Answer Sets* [51] demostrando una traducción del primer formalismo en TEL mediante la introducción de átomos auxiliares.

D.3 Conclusiones y trabajo futuro

La principal dificultad de este programa de investigación está relacionada con la variedad de resultados y métodos provenientes de campos tan diversos como la lógica modal y RNM. Por ejemplo, algunos resultados teóricos, como i han sido han sido planteados al inicio de esta tesis pero la respuesta ha sido encontrada recientemente y después de un profundo estudio de trabajos relacionados en ambas áreas. De forma similar, iv constituye un trabajo reciente, aún no publicado, desarrollado durante mi estancia en el IRIT (Universidad de Toulouse), con la colaboración invalorable de Philippe Balbiani. Además, la implementación de las diferentes herramientas fruto de esta tesis ha requerido también un importante esfuerzo de combinación de herramientas desarrolladas para ASP, software para LTL y manejo de autómatas.

Aunque se ha realizado mucho trabajo, existen aún muchos temas abiertos que se pueden estudiar en el futuro. A continuación describimos brevemente los más destacables.

- Un primera tarea futura consiste en completar el trabajo sobre la axiomatización de THT para probar que ésta caracteriza completamente narrativas lineales. Hasta ahora hemos conseguido probar que el operador "necesario" cubre el cierre transitivo del operador "siguiente", esto es, que siempre existe un mundo accesible, pero puede no ser único. Esta axiomatización puede ser extendida, considerando los operadores "until" y "release".

- Otra futura línea de investigación interesante es ver si alguno de estos operadores modales es representable en términos de otros operadores o no. Es sabido que, en LTL, el operador "until" puede definirse en función del operador "release" y viceversa. Lamentablemente en el caso de THT (debido a que la doble negación no puede eliminarse en el caso general) esta característica no es tan obvia. Queda por lo tanto determinar en este caso si los operadores "until" y "release" guardan una relación similar.

- Aunque hemos demostrado que una de las direcciones del teorema de Kamp también se cumple entre THT y Quantified Equilibrium Logic (QHT), la versión primer orden que Equilibrium Logic, realmente ignoramos si la otra dirección se cumple o no. Esta dirección consiste en determinar si cualquier fórmula en QHT con una relación lineal entre las variables, denotada por "$<$", se puede representar en TEL o no.

- Siguiendo pasos similares a los realizados en TEL, se podrían explorar otras aproximaciones híbridas para el razonamiento temporal. Por ejemplo, [4] ha considerado una combinación de Dynamic LTL (DLTL) y Equilibrium Logic. De forma similar, otras combinaciones de Equilibrium

Logic y lógicas modales temporales como CTL o CTL* se pueden considerar. Otra posibilidad sería considerar diferentes aproximaciones de RNM o programación lógica tales como Well-Founded Semantics [118] o Completion [27].

- En el campo de la implementación, una posibilidad que ha sido explorada en un inicio pero considerada posteriormente como trabajo futuro es la utilización de herramientas no basadas en autómatas. Una posible herramienta a considerar sería `TSPASS` [41] (basada en resolución temporal) y que suponemos podría mejorar la eficiencia de las herramientas actuales a la hora de calcular modelos temporales de equilibrio.

- Finalmente, una linea de investigación interesante consistiría en explotar los resultados obtenidos para su aplicación en el campo de los lenguajes de acciones, proporcionando traducciones de los formalismos de acciones que están fuertemente ligados a formalismos ASP como, por ejemplo, $\mathcal{ALM}$ [62] o MAD [80]. De forma similar otro posible campo de aplicación de este formalismo podría ser la aplicación de TEL al campo de *AI planning*, especialmente para comparar el comportamiento de TEL con respecto a las aproximaciones actuales a *planning*, las cuales utilizan restricciones temporales como reglas heurísticas que mejoran su eficiencia.