Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

A resolution-based E-connected calculus

Lucas de Moura Amaral

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Orientadora
Prof.a Dr.a Cláudia Nalon

Brasília
2019

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

```
        Amaral, Lucas de Moura
A485r      A resolution based E-connected calculus / Lucas de Moura
        Amaral; orientador Cláudia Nalon. -- Brasília, 2019.
           60 p.


           Dissertação (Mestrado - Mestrado em Informática) --
        Universidade de Brasília, 2019.


           1. modal logics. 2. connections. I. Nalon, Cláudia,
        orient. II. Título.
```

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

A resolution-based E-connected calculus

Lucas de Moura Amaral

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Brasília, 25 de fevereiro de 2019

Dedicatória

Dedico este trabalho à memória do meu pai, Gesley, que sempre fez de tudo para me dar uma boa educação e me incentivou a escolher o curso de Ciência de Computação. Também à minha família, que sempre me ajuda em tudo que preciso e não deixa de demonstrar interesse em minha vida: à minha mãe, Esther; meu padastro, Márcio; e aos meus irmãos Thiago, Davi e Benjamin.

Acknowledgements

I thank the Computer Science department and my lecturers for providing a great background, both in technical and theoretical aspects. In particular, I thank my supervisor, Cláudia, for her exceptional availability and understanding, both of which were essential for conclusion of this work. I also give an special acknowledgement to CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) for their support, funding a period of my work.

Abstract

We introduce a calculus to reason about $\mathcal{E}$-connections, which provide a computationally robust method to combine arbitrary *Abstract Description Systems* (ADSs). ADSs, introduced by Baader et al, are a generalization of various logics such as temporal, spatial, epistemic description and modal logics in general. In this work, we restrict the logics to be combined to normal modal logics. We provide a resolution-based calculus to deal with connections, assuming that the global satisfiability problems of the component logics are decidable. This allows us to focus on reasoning only about the restrictions imposed by the $\mathcal{E}$-connections, leaving domain-specific reasoning to the component logic.

One of the most important steps required to achieve this is the proper separation of syntactical elements related to different components via a proposed normal form. Therefore, we provide the full set of transformation rules, presenting proofs for termination and preservation of satisfiability of this transformation.

This work presents the correctness, completeness and termination proofs for the proposed calculus. We also make available a proof-of-concept implementation, based on the K$_{S}$P prover. We discuss the results of the evaluation and suggest some modifications that can be made to improve performance, paving the way for the development of a future modular and efficient implementation.

Keywords: modal logic, combination of logics, connections, automated reasoning, resolution

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Propositional modal languages are extensions of the propositional language with additional operators to express modalities. A modality is an expression used to qualify the truth of a sentence. As an example, we can qualify the sentence “It rains” using the possibility modality. Thus, we get the new sentence “It is possible that it rains”. Such modalities are expressed through *modal operators*.

Modal and other non-classic logics are used to formalise and reason about different notions, as spatial requirements, temporal aspects, and others. Because of its diversity and expressivity, such logics are object of interest in areas such as Knowledge Representation (KR) and Ontologies [4], Model Checking [12] and Multi-agent systems [11]. Applications in these areas usually talk about several different notions at once, combining spatial and temporal formalisms, different types of descriptive reasoning and even epistemic aspects.

Such formalisms are normally difficult to extend and, consequently, methods for combining different logics were developed, such as products [18], fusions [19] and fibrings [8]. The method of connections or $\mathcal{E}$-connected logics [9], in particular, has gained attention in the KR community, mainly for its intuitive semantics, generality and robustness regarding decidability preservation. It was originally proposed as a method for combining abstract description systems (ADS), which are a generalisation of description and modal logics, in such a way that, if the components are decidable, the resulting system is still decidable, although it may have an increase in complexity [9].

This method is closely related to Distributed Description Logics (DDL), but has greater expressivity [9]. DDLs provide a formalism for reasoning with multiple description logics, with interaction of different ontologies being *unidirected*, in the sense that the domain A maps the information of the domain B into their own context, having limited impact on the knowledge of the domain B [3]. $\mathcal{E}$-connections are more flexible in the sense that they provide a method for building new complex concepts from the basic concepts of the components. In the context of modal logic, this allow us to construct formulae using modalities of other logics through the use of *bridge modalities*.

An interesting aspect of $\mathcal{E}$-connections is that they do not, by themselves, add properties such as confluence, which may cause loss of decidability [9]. This occurs, for example, in approaches for combinations based on products [7]. Because $\mathcal{E}$-connections keep the languages disjoint, such behaviour does not occur. To obtain such properties, they must be explicitly added as axioms.

To the best of our knowledge, no dedicated automated reasoning solutions have been developed for this type of combination of logics. In [16], a simplified version of $\mathcal{E}$-connections was presented, together with a calculus for reasoning about such connection that is correct, complete and terminating. However, the calculus was restricted to local reasoning and the combined logics were fixed. In this work, we propose a calculus that extends the one in [16] for global reasoning, while taking a modular approach for the component logics. This is done by syntactically separating elements related to different components, which allow us to develop rules that focus on manipulating only the connection modalities. For handling domain-specific reasoning, before applying a rule we check if the calculus for the component can derive some necessary clauses given the current knowledge base.

1.2 Outline

This work presents a formal calculus for the $\mathcal{E}$-connections of modal logics, which manipulate only the formalisms regarding the links between the logics and leaves the domain-specific reasoning for the particular component logics being combined. This presentation is done in the following way:

1. Chapter 2 presents the basic definitions regarding modal logic and formal calculi, exemplified through the classical modal logic $\mathsf{K}_{(n)}$ and reviewing the concepts of termination, correctness, completeness.

2. Chapter 3 discusses some formalisms regarding combinations of logics, including the one used in this work, $\mathcal{E}$-connections.

3. Chapter 4 presents our main results. We introduce the calculus for $\mathcal{E}$-connections, including an appropriate normal form and the related definitions regarding the $\mathcal{E}$-connected logic problem and satisfiability. This chapter also presents an overview of our prototypical implementation and its experimental evaluation.

4. Chapter 5 contains all the theoretical results regarding the normal form and the calculus, including correctness, completeness and termination.

5. Chapter 6 concludes this text, making a brief review of the results and discusses future work.

Chapter 2

Modal logic and basic logic notions

In this chapter we give the basic theory necessary to the understanding of this work. We focus mainly in two aspects: basic modal logics and formal calculi.

2.1 The normal modal logic $\mathsf{K}_{(n)}$

Propositional modal languages extend the classic propositional language with *modal operators* which are capable of expressing different modalities of *truth*. A modal language with more than one primitive modal operator, that is, a operator that cannot be obtained through a combination of the others, is called a *multimodal language*. There are several different modal languages, for instance, conveying notions of time, that is, *when* something is true; notions of space, that is, *where* something is true; notions of belief, that is, if something is *believed* to be true, and many others [2]. The most common notions associated with modalities are those of necessity and possibility.

One of the most basic modal languages is obtained through an extension of the classic propositional language by two modal operators: one representing necessity ($[a]$) and the other, possibility ($\langle a\rangle$). We may exemplify the sentences that this language can express by considering the formulae $\langle a\rangle\neg p$ and $[b]p$, where p denotes the fact that "the sky is blue” and $\neg p$ denotes its negation. Then, the first formula expresses the sentence "Agent a considers that it is possible that the sky is not blue”, and the second formula means "Agent b considers that it is necessary that the sky is blue”.

Both these sentences may be true at the same time through reasoning in different contexts, called *possible worlds*. The meaning of each modal operator is associated with an accessibility relation over the possible worlds. Then, given a world w, an agent may reason about the truth of a proposition by examining the information of the worlds accessible through the agent’s relation from the world w.

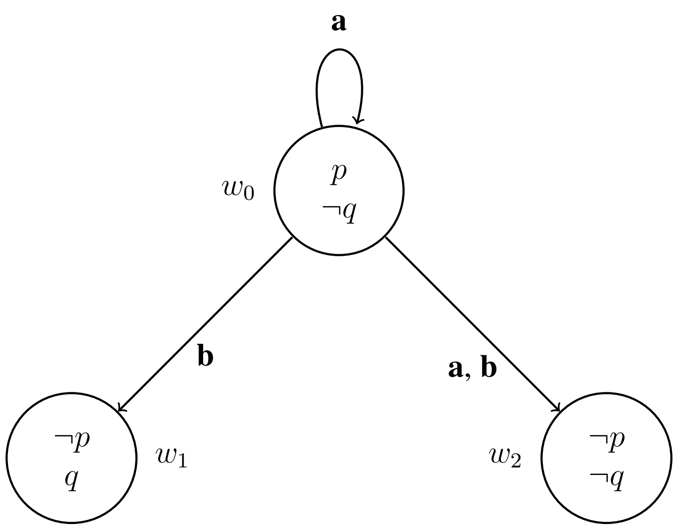


Figure 2.1: Example of contextual reasoning

Fig. 2.1 shows an exemple of *contextual reasoning*. In this and in the following examples, we use circles to represent worlds, labelled edges represent the agent's relations and the (negated) propositional symbols inside the circles represent what is true at that world. Agent a, reasoning from world w_0, considers $\neg q$ necessary, as at every world that is accessible from w_0 through the relation **a**, $\neg q$ holds. Meanwhile, agent b considers q possible, as q holds at the world w_1, which is accessible from w_0 through the relation **b**.

The modal operators enrich the language, as we may now talk not only about a single absolute truth in a system, but we may refer to different contexts and possibilities for truth. In the following subsections we will present the syntactical and semantic formalisms regarding the basic modal logic $\mathsf{K}_{(n)}$. The definitions given in the next sections follow standard presentations of modal logics [2] as well as the one given in [16].

We will refer to a *logic* L as the combination of its syntax $\mathcal{F}(\mathsf{L})$ together with an appropriate semantics, given by the relation $\models_{\mathsf{L}}$.

2.1.1 Syntax

Formulae are built from a countable set of propositional symbols, $\mathcal{P} = \{p, q, p', q', ...\}$. The finite set of *agents* is defined as $\mathcal{A} = \{1, ..., n\}$, $n \in \mathbb{N}$. We extend the set of propositional connectives ($\mathbf{true}, \neg, \wedge$) with the unary modal operators $\mathcal{M} = \{ \boxed{1}, ..., \boxed{n} \}$, where $\boxed{a}\,\varphi$, $a \in \mathcal{A}$, is read as "agent a considers φ necessary". When $n = 1$, we may omit the index. The fact that agent a considers φ possible is denoted by $\langle a \rangle\varphi$, which is an abbreviation for $\neg\,\boxed{a}\neg\varphi$. The language of $\mathsf{K}_{(n)}$ is defined over the alphabet $\mathcal{P} \cup \mathcal{M} \cup \{\mathbf{true}, \neg, \wedge\}$:

Definition 1. The *set of well-formed formulae*, $\mathcal{F}(\mathsf{K}_{(n)})$, is defined inductively:

- if $p \in \mathcal{P}$, then $p \in \mathcal{F}(\mathsf{K}_{(n)})$
- **true** $\in \mathcal{F}(\mathsf{K}_{(n)})$
- if φ and ψ are in $\mathcal{F}(\mathsf{K}_{(n)})$, then:
  - $\neg\varphi \in \mathcal{F}(\mathsf{K}_{(n)})$
  - $(\varphi \wedge \psi) \in \mathcal{F}(\mathsf{K}_{(n)})$
  - $\boxed{a}\varphi \in \mathcal{F}(\mathsf{K}_{(n)}),\ \forall a \in \mathcal{A}$

Formulae of the form $(\varphi \vee \psi)$ and $(\varphi \to \psi)$ are abbreviations for $\neg(\neg\varphi \wedge \neg\psi)$ and $\neg\varphi \vee \psi$, respectively. The symbol **false** is an abbreviation for $\neg$**true**. If there is only one agent in the logic being considered, we may omit the index inside the modal operator.

Example 1. Let φ and ψ be formulae of the basic modal language. The following are considered well-formed formulae: $(\varphi \wedge \neg\varphi)$, $\boxed{a}(\varphi \vee \psi)$, $(\varphi \to (\psi \vee \neg\psi))$. In contrast, the following expressions are ill-formed: $\neg \wedge \psi$, $\varphi\ \boxed{a}$, $\varphi\vee \to \psi$.

Definition 2. We call *(propositional) literals* all formulae of the form p or $\neg p$, for any propositional symbol p in $\mathcal{P}$. *Modal literals* are Formulae of the form $\boxed{a}p$, $\boxed{a}\neg p$, $\langle a\rangle p$, $\langle a\rangle\neg p$, for $a \in \mathcal{A}$.

Definition 3. We denote the set of *propositional literals* of a formula φ by $\mathcal{L}it(\varphi)$. We also extend this notation to a set $\mathcal{S}$ of formulae by writing $\mathcal{L}it(\mathcal{S})$, that is, $\mathcal{L}it(\mathcal{S}) = \bigcup_{\varphi\in\mathcal{S}} \mathcal{L}it(\varphi)$.

2.1.2 Semantics

To give meaning to a formula and be able to talk about sentences that are logical consequence of a set of statements, we need to formalise the semantics of the logic. That is, we need to define what is a model for the basic modal language. This is done via *Kripke structures*.

Definition 4. A *Kripke structure* $\mathbb{M}$ *for* n *agents over* $\mathcal{P}$ is a tuple $\mathbb{M} = \langle \mathcal{W}, \pi, \mathcal{R}_1, ..., \mathcal{R}_n\rangle$, where $\mathcal{W}$ is a non-empty set of possible worlds; the function $\pi(w) : \mathcal{P} \to \{true, false\}$, $w \in \mathcal{W}$, is an *interpretation* that gives each world in $\mathcal{W}$ a valuation to propositions; and, for all $a \in \mathcal{A} = \{1, ..., n\}$, $\mathcal{R}_a \subseteq \mathcal{W} \times \mathcal{W}$ is a binary relation over $\mathcal{W}$. Truth of a formula at a world w is given by:

- $(\mathbb{M},\ w) \models_\ell$ **true**
- $(\mathbb{M},\ w) \models_\ell p$ iff $\pi(w)(p) = true$, where $p \in \mathcal{P}$
- $(\mathbb{M},\ w) \models_\ell \neg\varphi$ iff $(\mathbb{M},\ w) \not\models_\ell \varphi$
- $(\mathbb{M},\ w) \models_\ell (\varphi \wedge \psi)$ iff $(\mathbb{M},\ w) \models_\ell \varphi$ and $(\mathbb{M},\ w) \models_\ell \psi$
- $(\mathbb{M},\ w) \models_\ell \boxed{a}\,\varphi$ iff for all w' such that $(w,\ w') \in \mathcal{R}_a,\ (\mathbb{M},\ w') \models_\ell \varphi$

Let $\mathbb{M} = \langle \mathcal{W}, \pi, \mathcal{R}_1, ..., \mathcal{R}_n \rangle$ be a Kripke structure.

Definition 5. We say that a formula φ is *(locally) satisfiable in a model* $\mathbb{M}$, denoted $\mathbb{M} \models_\ell \varphi$, if there exists a world $w \in \mathbb{M}$ such that $(\mathbb{M},\ w) \models_\ell \varphi$. In this case, $\mathbb{M}$ is called a *model* for φ. We extend this notion to sets in the following way: Let $\Gamma \subseteq \mathcal{F}(\mathsf{K}_{(n)})$, $\mathbb{M}$ a model and w a world in $\mathbb{M}$. If $(\mathbb{M}, w) \models_\ell \varphi$ for each $\varphi \in \Gamma$, then $(\mathbb{M}, w) \models_\ell \Gamma$, and $\mathbb{M}$ is a model for Γ.

Definition 6. We say that φ (resp. Γ) is *satisfiable* if there exists a model $\mathbb{M}$ such that φ (resp. Γ) is satisfiable in the model $\mathbb{M}$.

Example 2. Fig. 2.2 represents the Kripke structure $\mathbb{M} = \langle \mathcal{W}, \pi, \mathcal{R} \rangle$, where $\mathcal{W} = \{w_0, w_1, w_2\}$; $\pi(w, p) = true$ at $w = w_0$, and $false$ otherwise; and $\pi(w, q) = true$ at $w = w_1$, and $false$ otherwise. The relation over worlds is $\mathcal{R} = \{(w_0, w_1), (w_0, w_2), (w_1, w_1), (w_2, w_2)\}$. The formulae $\Box\neg p$, p, $\Diamond q$ and $\Diamond\neg q$ are satisfiable at w_0, $\Box q$ is satisfiable at w_1 and $\Box\neg q$ is satisfiable at w_2.

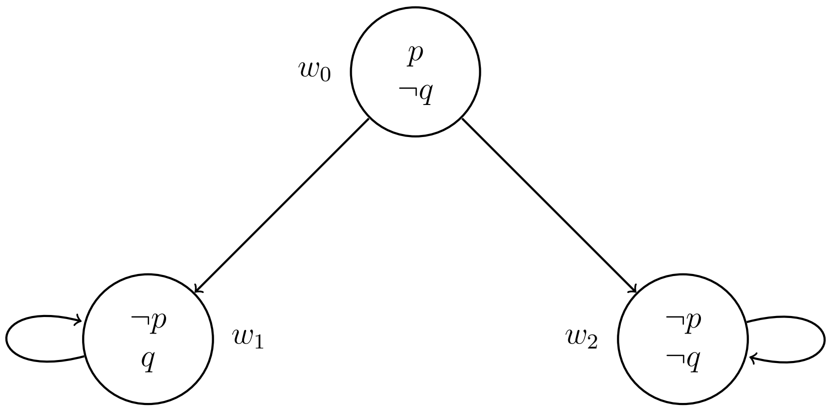


Figure 2.2: Graphical representation of the model $\mathbb{M}$

The above definitions give a notion of local satisfiability of a modal formula. We may also define a global satisfiability notion:

Definition 7. We say that a model $\mathbb{M}$ *globally satisfies* φ, denoted by $\mathbb{M} \models_g \varphi$ if, and only if, for all worlds w in $\mathcal{W}$, we have $(\mathbb{M},\ w) \models_\ell \varphi$.

Definition 8. We say that a formula φ is *globally satisfiable* if there is a model $\mathbb{M}$ such that $\mathbb{M} \models_g \varphi$.

Example 3. Fig. 2.3 represents the Kripke structure $\mathbb{M}' = \langle \mathcal{W}, \pi, \mathcal{R} \rangle$, where $\mathcal{W} = \{w_0, w_1\}$; p is $true$ everywhere; and q holds at w_0, but is $false$ at w_1. The relation over worlds is given by $\mathcal{R} = \{(w_0, w_1)\}$. The formulae $\Box p$ and p are globally satisfiable, but the formulae $\Diamond p$ and q are not.

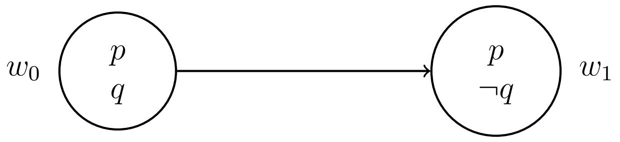


Figure 2.3: Graphical representation of the model $\mathbb{M}'$

Definition 9. Let $\mathcal{F}(\mathsf{K}_{(n)})$ be the language of $\mathsf{K}_{(n)}$, that is, its set of well-formed formulae. Let $\Gamma \subseteq \mathcal{F}(\mathsf{K}_{(n)})$ and $\varphi \in \mathcal{F}(\mathsf{K}_{(n)})$. We say that φ is a logical consequence of Γ, denoted $\Gamma \models_{\mathsf{K}_{(n)}} \varphi$, if every model of Γ is also a model of φ.

Definition 10. We say that a formula φ is *valid* if it is a logical consequence of the empty set, that is, if $\varnothing \models_{\mathsf{K}_{(n)}} \varphi$. This is also denoted as $\models_{\mathsf{K}_{(n)}} \varphi$.

Example 4. Let $\Gamma = \{p \to q, p\}$. Let $\mathbb{M}$ be a Kripke structure that satisfies Γ, that is, there is a world $w \in \mathbb{M}$ such that $(\mathbb{M}, w) \models_{\mathsf{K}_{(n)}} \Gamma$. Then, we have that $(\mathbb{M}, w) \models_{\mathsf{K}_{(n)}} p \to q$ and $(\mathbb{M}, w) \models_{\mathsf{K}_{(n)}} p$. By the semantics of implication, $(\mathbb{M}, w) \models_{\mathsf{K}_{(n)}} q$. Then, as $\mathbb{M}$ is arbitrary, q is a logical consequence of Γ.

2.2 Some notions about calculi

In this work, we propose a calculus to reason about a specific method of combination of logics. Then, we need to properly state some useful definitions about formal calculi and, specifically, about resolution, as our calculus, presented in Chapter 4, is based on this particular proof method. We denote by $2^{\mathcal{S}}$ the powerset of $\mathcal{S}$.

Definition 11. Let L be a logic. A *deductive calculus* is a pair $\langle \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{A} \subseteq \mathcal{F}(\mathsf{L})$ is a set of axioms and $\mathcal{R} \subseteq 2^{\mathcal{F}(\mathsf{L})} \times \mathcal{F}(\mathsf{L})$ is a set of inference rules.

Definition 12. Let $C = \langle \mathcal{A}, \mathcal{R} \rangle$ be a deductive calculus for a logic L, $\varphi \in \mathcal{F}(\mathsf{L})$ and $\Gamma \subseteq \mathcal{F}(\mathsf{L})$. A *proof for* φ *from* Γ is a sequence of formulae $(\varphi_0, \varphi_1, ..., \varphi_k)$, $\varphi_i \in \mathcal{F}(\mathsf{L})$, $0 \leq i \leq k$, where $\varphi_k = \varphi$ and each φ_i is either an axiom, a formula in Γ, or was obtained from the application of one of the inference rules to previous formulae in the sequence. We denote that $C = \langle \mathcal{A}, \mathcal{R} \rangle$ *proves* φ from Γ by $\Gamma \vdash_{\mathsf{L}}^{C} \varphi$. If $\Gamma = \varnothing$, we may simply write $\vdash_{\mathsf{L}}^{C} \varphi$.

We use deductive calculi to infer formulae from a set of assumptions. As such, we want to be able to show that (1) everything that we may deduce using the calculus is, in fact, a logical consequence of the assumptions and (2) we can deduce everything that is a consequence of our hypothesis. These notions are expressed by the following two properties of deductive calculi: soundness and completeness.

Let $\varphi \in \mathcal{F}(\mathsf{L}), \Gamma \subseteq \mathcal{F}(\mathsf{L})$, and $C = \langle \mathcal{A}, \mathcal{R} \rangle$ be a deductive calculus.

Definition 13. We say that C is *strongly sound* if $\Gamma \vdash_{\mathsf{L}}^{C} \varphi$, then $\Gamma \models_{\mathsf{L}}^{C} \varphi$ holds.

Definition 14. We say that C is *strongly complete* if $\Gamma \models_{\mathsf{L}}^{C} \varphi$, then $\Gamma \vdash_{\mathsf{L}}^{C} \varphi$ holds.

In this work, we use a resolution-based approach to our calculus. Before properly introducing what resolution means in the context of propositional logic, we must define the notion of *clauses*.

Definition 15. A *clause* is a disjunction of literals. A single literal is a clause, called the *unit clause*. The constant **false** is also a clause, called the *empty clause*.

We say that a formula is in *Negated Normal Form* (NNF) if it only contains the connectives $\wedge$, $\vee$ and negations are only applied to propositional symbols. We say that a formula is in *Clausal Normal Form* (CNF) if it is a conjunction of clauses. Any set of formulae can be first transformed to NNF and then to CNF [13]. This transformation is necessary for clausal resolution-based calculi.

Resolution-based calculi were introduced in [17] as a method for testing the (un)satisfiability of a set of first-order clauses. These calculi have been highly used both for its simplicity and efficiency [10].

Definition 16. The resolution-based calculus for propositional logic is the pair $Res = \langle \emptyset, \mathcal{R} \rangle$, where $\mathcal{R}$ is singleton set containing the *binary resolution* rule:

$$\begin{array}{llll} [\textbf{RES}] & D & \vee & l \\ & D' & \vee & \neg l \\ \hline & D & \vee & D' \end{array}$$

where D and D' are clauses and l is a literal. The premises are called *parent clauses* and the conclusion is called *resolvent*. The literals l and $\neg l$ are called *complementary literals*. Two clauses can be *resolved* if they contain complementary literals.

This method is able to check the (un)satisfiability of a set of clauses through *saturation*. This means that, given a set Γ of propositional clauses, we check the satisfiability of this set by adding to the clause set the new formulae generated through the application of the binary resolution inference rule until either the empty clause is generated, or no new clauses can be found. Note that, as the number of literals occurring in Γ is finite, then so is the amount of new clauses that can be generated. This means that the method must terminate. If the empty clause is found, then, as the rule is sound, we know that **false** is a logical consequence of Γ, which means that Γ is unsatisfiable. If the empty clause is not found, then, as the method is complete, we know that the original set is satisfiable. For more details on the soundness and completeness of resolution for propositional logic, refer to [17].

Example 5. We show an example of the application of this method to show the unsatisfiability of a set of clauses, taken from [13]. Consider the set $\Gamma = \{\neg p \vee q, \neg r \vee q, \neg q \vee s, p, \neg s\}$. We generate new clauses through repeated application of the resolution rule, until we can derive **false**.

1. $\neg p \vee q$	
2. $\neg r \vee q$	
3. $\neg q \vee s$	
4. p	
5. $\neg s$	
6. $\neg p \vee s$	$[RES, 1, 3]$
7. q	$[RES, 1, 4]$
8. $\neg r \vee s$	$[RES, 2, 3]$
9. $\neg q$	$[RES, 3, 5]$
10. **false**	$[RES, 7, 9]$

Then, as **false** is generated by the resolution calculus, the original set is unsatisfiable. ■

Chapter 3

Combinations of logics

This chapter introduces the concept of combinations of logics, briefly mentioning some usual forms of combining logics and presenting the method that is the focus of this work, the $\mathcal{E}$-connections method.

3.1 Methods of combinations

In the past decades, a great variety of logics were developed to represent various aspects of objects, such as spatial, temporal and descriptive aspects of such objects. However, given a specific domain to be represented, even with this great diversity of logics there is a considerable chance that there is no logic that is capable of represent all interesting aspects of such objects, or that it does so by becoming too complex.

For these reasons, there has been extensive research in developing formalisms to combining different logics. Such a formalism is capable of, given two logics, construct another one, with models being derived from the models of each component, and capable of syntactically represent elements of each language, with some degree of interaction. The restrictions on the models, the syntactic elements and their interaction are given by the specific formalism used to combine the logics.

One of the simplest methods for combining logics is called *fusion* [7]. Given two propositional modal logics L_1 and L_2, with disjunct sets of modal operators $\mathcal{M}_1 = \{\ \boxed{1}_1, ..., \ \boxed{n}_1\}$ and $\mathcal{M}_2 = \{\ \boxed{1}_2, ..., \ \boxed{m}_2\}$, the *fusion* of L_1 and L_2, denoted by $\mathsf{L}_1 \otimes \mathsf{L}_2$, is the propositional modal logic with the set of modal operators $\mathcal{M} = \mathcal{M}_1 \cup \mathcal{M}_2$. Let $\mathbb{M}_1 = \langle \mathcal{W}, \pi_1, \mathcal{R}_1^1, ..., \mathcal{R}_n^1 \rangle$ be a model for L_1 and $\mathbb{M}_2 = \langle \mathcal{W}, \pi_2, \mathcal{R}_1^2, ..., \mathcal{R}_m^2 \rangle$ be a model for L_2. A *model* for $\mathsf{L}_1 \otimes \mathsf{L}_2$ is simply the Kripke structure $\mathbb{M} = \langle \mathcal{W}, \mathcal{R}_1^1, ..., \mathcal{R}_n^1, \mathcal{R}_1^2, ..., \mathcal{R}_m^2, \pi \rangle$, where the set of worlds $\mathcal{W}$ is shared by both logics.

Example 6. Consider the fusion of $\mathsf{K}_{(n)}$ with the linear temporal logic, **LTL**. **LTL** is a propositional modal language with modalities referring to time. It has two basic modal operators,

$\mathcal{X}$ (ne**x**t) and $\mathcal{U}$ (**u**ntil). Additional modal operators may be defined based on these, such as $\mathcal{G}$ (always) and $\mathcal{F}$ (eventually). The semantics of **LTL** is based on a discrete, linear notion of time, which means there is only one possible future. This means that, with these new operators, we can say that a formula φ is true at the next instant of time through the sentence $\mathcal{X}\varphi$, or that some formula φ is true until a condition ψ is met, $\varphi\mathcal{U}\psi$.

We may combine $\mathsf{K}_{(n)}$ and **LTL** to obtain a logic $\mathsf{K}_{(n)} \otimes$ **LTL** capable of expressing a change on what the agents consider possible or necessary through time. In this language, we may express sentences such as "agent a does not consider φ necessary until agent b does" $((\neg \boxed{a}\ \varphi)\ \mathcal{U}(\ \boxed{b}\varphi))$ or "agent a will eventually consider φ necessary" $(\mathcal{F}\ \ \boxed{a}\varphi)$. ■

The method of fusions is interesting both for its simplicity and because it usually preserves interesting properties of the component logics, such as Kripke completeness and the finite model property [7]. Another interesting aspect of fusions is their computational robustness, in the sense that they usually preserve decidability[1].

Another usual method for combining logics is the *product* of logics. The language of the product $\mathsf{L}_1 \times \mathsf{L}_2$ is, again, simply the basic propositional language with modalities from both logics. The difference is how we combine the models. For simplicity, we consider that the model of each logic has only one relation each. For example, given the models $\mathbb{M}_1 = \langle \mathcal{W}_1, \mathcal{R}_1, \pi_1 \rangle$ and $\mathbb{M}_2 = \langle \mathcal{W}_2, \mathcal{R}_2, \pi_2 \rangle$, the model for the product $\mathsf{L}_1 \times \mathsf{L}_2$ is defined as $\mathbb{M} = \langle \mathcal{W}_1 \times \mathcal{W}_2, \bar{\mathcal{R}}_h, \bar{\mathcal{R}}_v, \pi_1 \times \pi_2 \rangle$, where, for $w_1, w_2 \in \mathcal{W}_1$ and $v_1, v_2 \in \mathcal{W}_2$, we have:

- $\langle w_1, v_1 \rangle \bar{\mathcal{R}}_h \langle w_2, v_2 \rangle$ if, and only if, $w_1 \mathcal{R}_1 w_2$ and $v_1 = v_2$;
- $\langle w_1, v_1 \rangle \bar{\mathcal{R}}_v \langle w_2, v_2 \rangle$ if, and only if, $w_1 = w_2$ and $v_1 \mathcal{R}_2 v_2$;
- $(\pi_1 \times \pi_2)(\langle w_1, v_1 \rangle)(p) = true$ if, and only if, $\pi_1(w_1)(p) = true$ and $\pi_2(v_1)(p) = true$

Products of logics are natural constructs which allow representing combinations of aspects of the real world, through interactions of modal operators related to time and space, for example. However, it is worth noting that by using products, we may include properties such as commutativity of operators (for instance, $\langle 1 \rangle \langle 2 \rangle p \to \langle 2 \rangle \langle 1 \rangle p$), which may cause loss of decidability of the combination [7].

Although fusions and products are well studied and have a natural geometric interpretation, other formalisms for combining logics have been proposed, trying to better control the interactions provided by the formalism whilst also maintaining expressivity. The method of $\mathcal{E}$-connections is one such formalism, which is introduced in the next subsection.

[1]As a very well-know example that fusions do not preserve all desirable properties of the component logics, we note that the propositional dynamic logic (**PDL**) enriched with the global modality (K_n^*) is highly undecidable whilst both **PDL** and K_n^* are decidable. See [2, Section 6.5].

3.2 Connections

Let L_1 and L_2 be disjoint multimodal normal logics that are to be connected. Before defining the language of a modal connection, we must define the set of connecting modal operators:

$$\mathcal{M} = M_1 \cup M_2, \text{ with } M_1 = \{\langle j \rangle^1 \mid j \in \mathcal{I}_1\} \text{ and } M_2 = \{\langle k \rangle^2 \mid k \in \mathcal{I}_2\}$$

where $\mathcal{I}_1$ and $\mathcal{I}_2$ non-empty countable set of indexes.

The set of formulae of the connected modal language $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$ may be split in a set of *1-formulae* and a set of *2-formulae*. Intuitively, *1-formulae* are formulae from the logic L_1 enriched with new connecting modal operators to talk about *2-formulae*. The same goes for *2-formulae*:

Definition 17. The sets of *1-formulae* and *2-formulae* of $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$ are defined through simultaneous induction, with $i \in \{1,\ 2\}$:

- Every propositional symbol of L_i is an *i-formula*
- The set of *i-formulae* is closed under the Boolean and modal operators of L_i
- If φ is a *1-formula* and $k \in \mathcal{I}_2$, then $\langle k \rangle^2\ \varphi$ is a *2-formula*
- If ψ is a *2-formula* and $j \in \mathcal{I}_1$, then $\langle j \rangle^1\ \psi$ is a *1-formula*

When we are referring to some component $i \in \{1, 2\}$, the other component will be referred as $\bar{i}$. That is, $\bar{1} = 2$ and $\bar{2} = 1$.

Example 7. Let $p, q \in \mathcal{F}(\mathsf{L}_1)$ and $l, m \in \mathcal{F}(\mathsf{L}_2)$. Then, the expressions $p \wedge q$ and $\Box^1(l \to m)$ are 1-formulae, while the expressions $l \vee m$ and $\Diamond^2(p \vee q)$ are 2-formulae. ■

A *connected Kripke model* for $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$ consists of a Kripke model for L_1, a Kripke model for L_2 and an interpretation of the set $\mathcal{E}$ of relations associated with the connecting modal operators:

Definition 18. A connected Kripke model for the logic $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$ is the structure $\mathbb{M} = \langle \mathbb{W}_1,\ \mathbb{W}_2,\ (E^1_j)_{j \in \mathcal{I}_1},\ (E^2_k)_{k \in \mathcal{I}_2} \rangle$, where $\mathbb{W}_i$ is a Kripke model for L_i, $E^1_j \subseteq \mathcal{W}_1 \times \mathcal{W}_2$, for each $j \in \mathcal{I}_1$, and $E^2_k \subseteq \mathcal{W}_2 \times \mathcal{W}_1$, for each $k \in \mathcal{I}_2$. The members of the set $\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2$, with $\mathcal{E}_1 = \{E^1_j \mid j \in \mathcal{I}_1\}$ and $\mathcal{E}_2 = \{E^2_k \mid k \in \mathcal{I}_2\}$ are called *connecting relations*.

The *global satisfiability* of an *i-formula* of L_i is defined by extending the global satisfiability notion of the connected logics: Booleans and modalities of the logic L_i are defined as previously (see Chapter 2). The remaining cases are defined as follows. Let φ be an *1-formula*, and ψ a *2-formula*, $w_1 \in \mathcal{W}_1$ and $w_2 \in \mathcal{W}_2$. Then, the global satisfiability of the connected modalities is defined as follows.

Definition 19. Given a connected model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (E^1_j)_{j\in\mathcal{I}_1}, (E^2_k)_{k\in\mathcal{I}_2}\rangle$, the satisfiability of the connected modalities are as follows:

- $\mathbb{M} \models_g \Diamond_j^{\,1}\, \psi \iff \forall w_1 \in \mathcal{W}_1, \exists w_2 \in \mathcal{W}_2$ such that $w_1\, E^1_j\, w_2$, and $(\mathbb{M}, w_2) \models_\ell \psi$; and
- $\mathbb{M} \models_g \Diamond_k^{\,2}\, \varphi \iff \forall w_2 \in \mathcal{W}_2, \exists w_1 \in \mathcal{W}_1$ such that $w_2\, E^2_k\, w_1$, and $(\mathbb{M}, w_1) \models_\ell \varphi$

The method of $\mathcal{E}$-connections provides a controlled interaction between the languages, obtained through the *connecting operators* and their related *connecting relations*. This becomes clear in the following examples.

Example 8. Consider again the languages of $\mathsf{K}_{(n)}$ and **LTL**. We may combine both logics into the logic $\mathcal{C}^{\mathcal{M}}(\mathsf{K}_{(n)}, \mathbf{LTL})$, where the set $\mathcal{M} = \{\Diamond^{2}\}$ is the set of connecting modalities. We may express the statements "agent a will eventually know φ" through the sentence $\mathcal{F}\,\Box^{2}\,\boxed{a}\varphi$.

Example 9. Consider two models for the basic model logic, $\mathbb{M}_1 = \langle\{w_0, w_1\}, \{(w_0, w_1)\}, \pi_1\rangle$ and $\mathbb{M}_2 = \langle\{v_0, v_1, v_2\}, \{(v_0, v_1), (v_0, v_2), (v_1, v_1), (v_2, v_2)\}, \pi_2\rangle$; with $\pi_1(p) = \mathit{true}$ everywhere in $\mathbb{M}_1$ and $\pi_1(q) = \mathit{true}$ at w_0 only; $\pi_2(m) = \mathit{true}$ only at v_2 and $\pi_2(l) = \mathit{true}$ only at v_0.

We may connect these models with the relation $E^1 = \{(w_0, v_2), (w_1, v_1)\}$, obtaining the model for the connection, $\mathbb{M}'' = \langle\mathbb{M}_1, \mathbb{M}_2, \{E^1\}, \varnothing\rangle$. This model is graphically represented below, where the dashed lines show the connecting relations. We see that $\mathbb{M}'' \models_g \Box^{1}\neg l$, as for every world in $\mathbb{M}_1$, all worlds in $\mathbb{M}_2$ that are accessible satisfy $\neg l$. We also see that $\mathbb{M}'' \models_g (\Box^{1} m) \vee (\Box^{1}\neg m)$, as the worlds in $\mathbb{M}_1$ either only see worlds in $\mathbb{M}_2$ where m holds, or worlds where it does not.

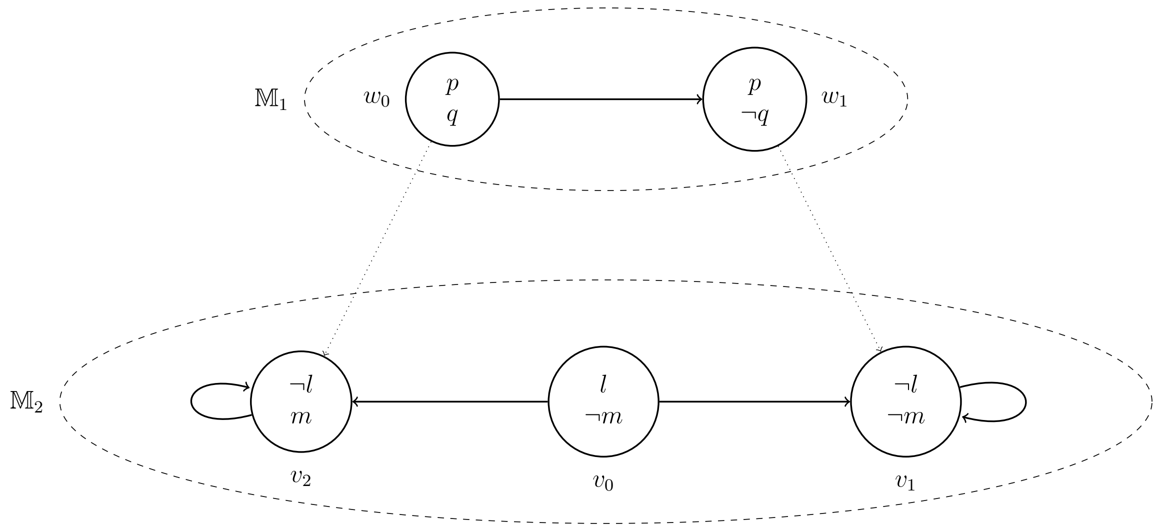


Figure 3.1: Graphical representation of the model $\mathbb{M}''$

Example 9 demonstrates how the languages of the connected logics are kept separated. Syntactically, interaction between the languages is done via the connecting operators, while the models are kept disjoint, which permit the evaluation of formulae to be done in an isolated way: i-formulae are evaluated in the i-component, and any $\bar{i}$-subformulae are evaluated in the opposing component, through the accessibility relations.

The characteristic of the method, which keeps the languages reasonably separated, allow us to propose, in this work, a calculus that can focus on reasoning only about the connections, leaving domain specific knowledge to the reasoners for each component. A possible implementation for such calculus, including the one shown in this work on Chapter 4, would mimic the separated nature of the models. Given a set of formulae with connecting modalities to test for satisfiability, such an implementation would *query* the reasoners of the component logics about the satisfiability of the formulae inside such modalities while transferring information between both reasoners.

Example 10. Let L_1 and L_2 be two logics to be connected, such that $l \in \mathcal{F}(\mathsf{L}_1)$ and $\varphi \in \mathcal{F}(\mathsf{L}_2)$. Fig. 3.2 exemplifies how an implementation of this calculus may be used to test the satisfiability of the set of 1-formulae $\Gamma = \{l \to \Diamond^{1}\varphi, l \to \boxed{\Box}^{1}\neg\varphi, l\}$. In the figure, the steps taken are numbered 1-6, with (1) being the initial problem. The $\mathcal{E}$-connected reasoner starts by querying the reasoner of Logic 2 to check if the set of 2-formulae $\{\varphi, \neg\varphi\}$ is satisfiable (2). The result is that this set is not satisfiable, which means that the left-hand side of the implications in $\{l \to \Diamond^{1}\varphi, l \to \boxed{\Box}\neg\varphi\}$, cannot hold (3). That is, $\neg l$ must be satisfiable. The $\mathcal{E}$-connected reasoner then has both 1-formulae l and $\neg l$, and it queries the reasoner of Logic 1 if the set $\{l, \neg l\}$ is satisfiable (4). As it answers that the set is unsatisfiable (5), we finally have that Γ is unsatisfiable (6).

■

In the next chapter we present a calculus based on these ideas, together with a normal form to keep the interaction between the languages to a minimum, facilitating the design of a calculus with characteristics of modular reasoning.

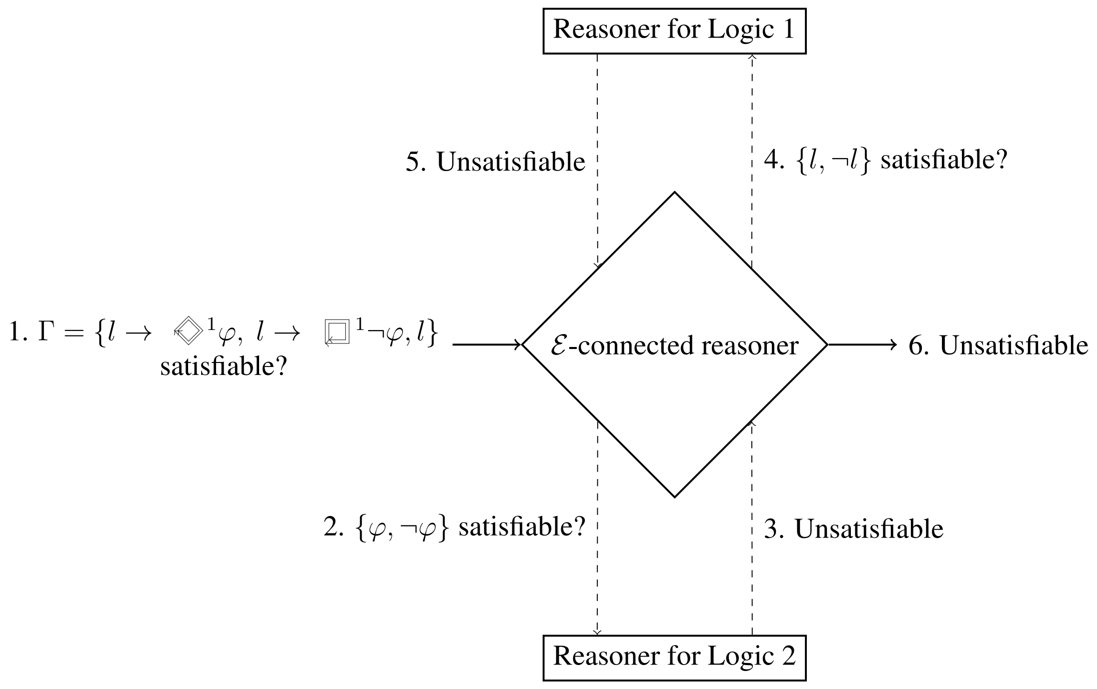

Reasoner for Logic 1
5. Unsatisfiable
4. $\{l, \neg l\}$ satisfiable?
1. $\Gamma = \{l \rightarrow \Diamond^1\varphi,\ l \rightarrow \Box^1\neg\varphi, l\}$ satisfiable?
$\mathcal{E}$-connected reasoner
6. Unsatisfiable
2. $\{\varphi, \neg\varphi\}$ satisfiable?
3. Unsatisfiable
Reasoner for Logic 2

Figure 3.2: Graphical representation of an implementation of the calculus

Chapter 4

$\mathcal{E}$-connected calculus

In this chapter we introduce the resolution-based calculus for reasoning about the connection of logics, $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$. This calculus operates on the so called $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, which is a structure containing the formulae that we wish to reason upon. For applying the inference rules of the calculus, we will require further that the problem is in a specific normal form, called *$\mathcal{E}$-connected normal form*, abbreviated as $\mathcal{E}_{conn}$-NF.

We first define what a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem is.

Definition 20. Let $\mathcal{G}_1,\ \mathcal{G}_2,\ \mathcal{K}_1,\ \mathcal{K}_2 \subset \mathcal{F}(\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2))$ be sets of formulae. A $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C}$ is a structure:

$$\langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$$

where each $\langle \mathcal{G}_i, \mathcal{K}_i\rangle$, for $i \in \{1, 2\}$, is called a partial (or component) $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$ problem.

The purpose of this work is to devise a method for deciding the satisfiability of a formula φ in the language of the connected logics. Instead of working on the formula itself, we carry out the satisfiability checking procedure over a correspondent equisatisfiable $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem. The satisfiability of $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problems is defined next.

Definition 21. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem and $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (E^1_j)_{j \in \mathcal{I}_1}, (E^2_k)_{k \in \mathcal{I}_2}\rangle$ be a connected Kripke model. The component $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ is satisfiable in $\mathbb{M}$, denoted by $\mathbb{M} \models_g \mathcal{C}_i$, if, and only if, for all $\varphi \in \mathcal{G}_i \cup \mathcal{K}_i$, we have that $\mathbb{W}_i \models_g \varphi$, where the definitions for satisfiability of connected modalities, adapted from Definition 19, are as follows:

- $\mathbb{M}_i \models_g \Diamond_{j}^{i}\, \psi$ if, and only if, $\mathbb{M}_{\bar{i}} \models_g \mathcal{C}_{\bar{i}}$ and, for all $w \in \mathcal{W}_i$, $\exists w' \in \mathcal{W}_{\bar{i}}$ such that $w\ E^i_j\ w'$ and $(\mathbb{M}_{\bar{i}}, w') \models_\ell \psi$.

Definition 22. We say that a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ is satisfiable if, and only if, there is a model $\mathbb{M} = \langle \mathbb{W}_1,\ \mathbb{W}_2, (E^1_j)_{j\in\mathcal{I}_1},\ (E^2_k)_{k\in\mathcal{I}_2}\rangle$ such that $\mathbb{M} \models_g \langle \mathcal{G}_1, \mathcal{K}_1\rangle$, that is, $\mathbb{M}$ satisfies the component $\langle \mathcal{G}_1, \mathcal{K}_1\rangle$.

We will always consider the satisfiability problem of 1-formulae. Thus, we consider that the $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ is satisfiable if, and only if, $\langle \mathcal{G}_1, \mathcal{K}_1\rangle$ is satisfiable. Note that our definitions do not require that the $\langle \mathcal{G}_2, \mathcal{K}_2\rangle$ is satisfiable, as a problem as $\langle\langle\{t_1\}, \{t_1 \to \boxed{j}^{1} t_2\}\rangle, \langle\{\mathbf{false}\}, \emptyset\rangle\rangle$ is satisfiable even though $\mathcal{G}_2 = \{\mathbf{false}\}$ cannot ever be satisfiable in any model for $\mathbb{C}_2$. However, definition Definition 21 makes it so that, if the 1-formulae force the existence of a 2-world in a model, then $\mathcal{G}_2 \cup \mathcal{K}_2$ must be globally satisfied in the model.

Before presenting the rules of the calculus, we present how to transform a formula into $\mathcal{E}_{conn}$-NF. This normal form is loosely based on the Separated Normal Form, presented in [6] and revisited in [16].

4.1 A normal form for connected logics

The idea is to apply a set of rewriting and renaming rules to obtain formulae in the appropriate clausal form. Rewriting is used to remove unwanted operators whilst the renaming rules replace complex formulae in the scope of modal operators by new propositional symbols until we have a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem with two disjoint pairs of sets of formulae, each pair belonging to the language of one of the logics. Moreover, at the end of the transformation, we will require that formulae in each of the sets of the corresponding $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem is in a specific form. The allowed forms of clauses are given in Table 4.1 (where l, l' are literals, $b, r \in \mathbb{N}$, $i \in \{1, 2\}$, $a \in \mathcal{A}^i$, where $\mathcal{A}^i$ is the set of modal agents of language L_i, and $k \in \mathcal{E}_i$). The separation between the languages is not complete, because in the scope of the connecting operators of the language L_i we still have literals in the language of $\mathsf{L}_{\bar{i}}$. However, we limit the occurrence of formulae inside the connecting operators to literals, and we place all the "connecting formulae" of L_i in the set $\mathcal{K}_i$.

Literal clause	$\bigvee_{b=1}^{r} l_b$
Positive a_i-clause	$l' \to \boxed{a}^{i} l$
Negative a_i-clause	$l' \to \langle a\rangle^{i} l$
Positive $\mathcal{E}_i^k$-clause	$l' \to \boxed{\boxed{k}}^{i} l$
Negative $\mathcal{E}_i^k$-clause	$l' \to \langle\langle k\rangle\rangle^{i} l$

Table 4.1: Form of clauses in $\mathcal{E}_{conn}$-NF

The definition of the *$\mathcal{E}$-connected normal form* is given below.

Definition 23. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem. We say that $\mathbb{C}$ is in $\mathcal{E}_{conn}$-NF if and only if, for $i \in \{1, 2\}$:

- all formulae in $\mathcal{G}_i$ are literal clauses.
- all formulae in $\mathcal{K}_i$ are a_i-clauses, that is, modal clauses of L_i, or $\mathcal{E}_i^k$-clauses, that is, connecting modal clauses.

We define next the set of transformation rules used to produce the normal form of a formula φ. At the end of the transformation, the i-formulae are placed in the appropriate $\mathcal{G}_i$ and $\mathcal{K}_i$ sets, with formulae that contain modalities belonging to $\mathcal{K}_i$ and formulae that do not contain modalities belonging to $\mathcal{G}_i$.

To ensure that the problem is in normal form at the end of the process, we apply the simplification and rewriting rules given in the Table 4.2 at any step of the transformation.

$\neg(\varphi \land \psi) \to \neg\varphi \lor \neg\psi$	$\neg(\varphi \lor \psi) \to \neg\varphi \land \neg\psi$
$\neg(\varphi \to \psi) \to \varphi \land \neg\psi$	$\neg\neg\varphi \to \varphi$
$\neg \boxed{a}^i \varphi \to \langle a \rangle^i \neg\varphi$	$\neg \boxed{k}^i \varphi \to \langle k \rangle^i \neg\varphi$
$\neg \langle a \rangle^i \varphi \to \boxed{a}^i \neg\varphi$	$\neg \langle k \rangle^i \varphi \to \boxed{k}^i \neg\varphi$
$(\varphi \lor \textbf{true}) \to \textbf{true}$	$\neg\textbf{false} \to \textbf{true}$
$(\varphi \land \textbf{false}) \to \textbf{false}$	$\neg\textbf{true} \to \textbf{false}$
$(\varphi \land \textbf{true}) \to \varphi$	
$(\varphi \lor \varphi) \to \varphi$	$(\varphi \land \varphi) \to \varphi$
$(\varphi \lor \neg\varphi) \to \textbf{true}$	$(\varphi \land \neg\varphi) \to \textbf{false}$

Table 4.2: Simplification and rewriting rules

Given an 1-formula $\varphi \in \mathcal{F}(\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2))$, we define the *initial* $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$*-problem*, also denoted by $start(\varphi)$, as $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\mathcal{K}_1 = \mathcal{G}_2 = \mathcal{K}_2 = \varnothing$ and the set $\mathcal{G}_1 = \{t_0, t_0 \to \varphi\}$ and t_0 is a literal not occurring in φ. We define $\mathcal{L}it(\Phi)$ as the set of literals occurring in the original formula φ.

Next, we give the rules to transform the problems into an appropriate clausal form. We first deal with the cases in which the formula on the right-hand side of the implication is a conjunction or a disjunction:

$$\begin{aligned}
\tau_\land :\quad & \langle \mathcal{G}_i \cup \{t \to \varphi_1 \land \varphi_2\}, \mathcal{K}_i\rangle \longrightarrow \langle \mathcal{G}_i \cup \{t \to \varphi_1, t \to \varphi_2\}, \mathcal{K}_i\rangle \\
\tau_\lor :\quad & \langle \mathcal{G}_i \cup \{t \to \varphi_1 \lor \varphi_2\}, \mathcal{K}_i\rangle \longrightarrow \langle \mathcal{G}_i \cup \{t \to \varphi_1 \lor t_1, t_1 \to \varphi_2\}, \mathcal{K}_i\rangle \\
& \qquad\qquad\qquad\qquad\qquad \text{where } \varphi_2 \text{ is not a disjunction of literals}
\end{aligned}$$

As the calculus to be presented in the next section is clausal, purely propositional formulae are required to be in clausal form, that is, they are disjunction of literals. The next rule transforms a formula $t \to \varphi$ into a disjunction of literals, when φ is already a propositional clause:

$$\tau_{\to}: \quad \langle \mathcal{G}_i \cup \{t \to \varphi\}, \mathcal{K}_i \rangle \longrightarrow \langle \mathcal{G}_i \cup \{\neg t \vee \varphi\}, \mathcal{K}_i \rangle$$
$$\text{where } \varphi \text{ is a disjunction of literals}$$

As previously said, during the transformation, the set $\mathcal{G}_i$ might contain formulae with occurrences of modalities. The following rule moves i-formulae which have modalities as main operators into the corresponding $\mathcal{K}_i$ set:

$$\tau_{move}: \quad \langle \mathcal{G}_i \cup \{t \to \varphi\}, \mathcal{K}_i \rangle \longrightarrow \langle \mathcal{G}_i, \mathcal{K}_i \cup \{t \to \varphi\} \rangle$$
$$\text{where the main operator in } \varphi \text{ is a modality}$$

The next rules rename complex subformulae occurring in the scope of modal operator (where t_1 is a new propositional symbol). The new definition for the subformula is moved to the appropriate $\mathcal{G}_i$ set (where φ is not a literal).

$$\begin{aligned}
\tau_{\Box}: &\quad \langle \mathcal{G}_i, \mathcal{K}_i \cup \{t \to \boxed{a}^{i} \varphi\} \rangle \longrightarrow \langle \mathcal{G}_i \cup \{t_1 \to \varphi\}, \mathcal{K}_i \cup \{t \to \boxed{a}^{i} t_1\} \rangle \\
\tau_{\Diamond}: &\quad \langle \mathcal{G}_i, \mathcal{K}_i \cup \{t \to \langle a \rangle^{i} \varphi\} \rangle \longrightarrow \langle \mathcal{G}_i \cup \{t_1 \to \varphi\}, \mathcal{K}_i \cup \{t \to \langle a \rangle^{i} t_1\} \rangle
\end{aligned}$$

Finally, we must deal with formulae in which there is an occurrence of a connecting modality. These transformation rules work in a similar way to the two previous rules, but we move the subformula being renamed to the global set of the other logic language. By doing so, we keep the vocabulary of the logics separated. Formulae in $\langle \mathcal{G}_i, \mathcal{K}_i \rangle$ have basically only the constructions of the language of L_i, with the exception of literals occuring in the scope of connecting modalities. Recall that $\bar{i}$ is the opposing index in $\{1, 2\}$. That is, $\bar{1} = 2$ and $\bar{2} = 1$.

$$\tau_{\boxed{\Box}}: \left\langle \begin{array}{c} \langle \mathcal{G}_i, \mathcal{K}_i \cup \{t \to \boxed{\boxed{a}}^{i} \varphi\} \rangle, \\ \langle \mathcal{G}_{\bar{i}}, \mathcal{K}_{\bar{i}} \rangle \end{array} \right\rangle \longrightarrow \left\langle \begin{array}{c} \langle \mathcal{G}_i, \mathcal{K}_i \cup \{t \to \boxed{\boxed{a}}^{i} t_1\} \rangle, \\ \langle \mathcal{G}_{\bar{i}} \cup \{t_1 \to \varphi\}, \mathcal{K}_{\bar{i}} \rangle \end{array} \right\rangle$$

$$\tau_{\langle\langle\rangle\rangle}: \left\langle \begin{array}{c} \langle \mathcal{G}_i, \mathcal{K}_i \cup \{t \to \langle\langle a \rangle\rangle^{i} \varphi\} \rangle, \\ \langle \mathcal{G}_{\bar{i}}, \mathcal{K}_{\bar{i}} \rangle \end{array} \right\rangle \longrightarrow \left\langle \begin{array}{c} \langle \mathcal{G}_i, \mathcal{K}_i \cup \{t \to \langle\langle a \rangle\rangle^{i} t_1\} \rangle, \\ \langle \mathcal{G}_{\bar{i}} \cup \{t_1 \to \varphi\}, \mathcal{K}_{\bar{i}} \rangle \end{array} \right\rangle$$

Note that in the above transformation rules, renaming occurs even if φ is a literal. However, to ensure termination of the transformation procedure, if φ is a literal, the rule is only applied if φ occurs in $\mathcal{L}it(\Phi)$, the set of literals occurring in the original formula.

We apply these transformation rules while it is possible to do so. Note that the cases where the transformation rules cannot be applied are three:

1. φ in $\mathcal{G}_i$ is a disjunction of literals, that is, φ is a literal clause;

2. $t \to \varphi$ is in $\mathcal{K}_i$ and φ is a modal literal, that is, $t \to \varphi$ is either a negative or a positive a_i-clause, with a single literal occuring inside the modality;
3. $t \to \varphi$ is in $\mathcal{K}_i$, φ is either of the form $\boxed{k}^{i} t$ or $\langle k \rangle^{i} t$, and $t \notin \mathcal{L}it(\Phi)$, that is, $t \to \varphi$ is either a negative or a positive $\mathcal{E}_i^k$-clause.

Thus, when the transformation procedure stops, the problem is into the $\mathcal{E}_{conn}$ normal form, as shown in Chapter 5.

We now give an example of an application of the transformation procedure.

Example 11. Consider the following 1-formula in NNF:

$$\varphi = (\neg q \vee \boxed{j}^{1}(l \wedge m)) \wedge (p \vee q)$$

where $\{p, q\}$ is in L_1, $\{l, m\}$ is in L_2. The initial problem is:

$$\left\langle \begin{array}{c} \langle \{t_0, t_0 \to (\neg q \vee \boxed{j}^{1}(l \wedge m)) \wedge (p \vee q)\}, \varnothing \rangle, \\ \langle \varnothing, \varnothing \rangle \end{array} \right\rangle$$

We then apply $\tau_\wedge$ to the conjunction in $\mathcal{K}_1$, obtaining:

$$\left\langle \begin{array}{c} \langle \{t_0, t_0 \to (\neg q \vee \boxed{j}^{1}(l \wedge m)), t_0 \to (p \vee q)\}, \varnothing \rangle, \\ \langle \varnothing, \varnothing \rangle \end{array} \right\rangle$$

Next, we apply $\tau_\to$ to $t_0 \to (p \vee q)$, as both p and q are literals, obtaining:

$$\left\langle \begin{array}{c} \langle \{t_0, t_0 \to (\neg q \vee \boxed{j}^{1}(l \wedge m)), \neg t_0 \vee p \vee q\}, \varnothing \rangle, \\ \langle \varnothing, \varnothing \rangle \end{array} \right\rangle$$

We then apply $\tau_\vee$ to $t_0 \to (\neg q \vee \boxed{j}^{1}(l \wedge m))$. Note that the propositional symbol t_1 is introduced to rename $(\boxed{j}^{1}(l \wedge m))$:

$$\left\langle \begin{array}{c} \langle \{t_0, t_0 \to (\neg q \vee t_1), \neg t_0 \vee p \vee q, t_1 \to \boxed{j}^{1}(l \wedge m)\}, \varnothing \rangle, \\ \langle \varnothing, \varnothing \rangle \end{array} \right\rangle$$

As $t_1 \to \boxed{j}^{1}(l \wedge m)$ in $\mathcal{G}_1$ contains a modality, we apply τ_{move} in order to move this formula to the correct set, $\mathcal{K}_1$:

$$\left\langle \begin{array}{c} \langle \{t_0, t_0 \to (\neg q \vee t_1), \neg t_0 \vee p \vee q\}, \{t_1 \to \boxed{j}^{1}(l \wedge m)\} \rangle, \\ \langle \varnothing, \varnothing \rangle \end{array} \right\rangle$$

Next, we rename $(l \wedge m)$ by t_1' by applying the $\tau_{\Box}$ transformation rule:

$$\left\langle \begin{array}{c} \langle \{t_0, t_0 \rightarrow (\neg q \vee t_1), \neg t_0 \vee p \vee q\}, \{t_1 \rightarrow \boxed{j}^{1} t'_1\}\rangle, \\ \langle \{t'_1 \rightarrow (l \wedge m)\}, \varnothing \rangle \end{array} \right\rangle$$

By an application of $\tau_\wedge$ we obtain the following $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem:

$$\left\langle \begin{array}{c} \langle \{t_0, t_0 \rightarrow (\neg q \vee t_1), \neg t_0 \vee p \vee q\}, \{t_1 \rightarrow \boxed{j}^{1} t'_1\}\rangle, \\ \langle \{t'_1 \rightarrow l, t'_1 \rightarrow m\}, \varnothing \rangle \end{array} \right\rangle$$

And, finally, the $\tau_\rightarrow$ transformation rule is applied to the three formulae which are not yet in the normal form. We obtain:

$$\left\langle \begin{array}{c} \langle \{t_0, \neg t_0 \vee \neg q \vee t_1, \neg t_0 \vee p \vee q\}, \{t_1 \rightarrow \boxed{j}^{1} t'_1\}\rangle, \\ \langle \{\neg t'_1 \vee l, \neg t'_1 \vee m\}, \varnothing \rangle \end{array} \right\rangle$$

which is in the $\mathcal{E}_{conn}$-NF.

Once a formula is transformed into a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem into $\mathcal{E}_{conn}$-NF, the resolution method for $\mathcal{E}$-connected logics can be applied. The calculus is introduced in the next section.

4.2 The calculus

We assume that the logics L_1 and L_2 are decidable logics, and that C_1 and C_2 are terminating, strongly sound and strongly complete calculi for L_1 and L_2, respectively. Having decision procedures for the component logics ensure us that we only need to provide inference rules for the connection between them. The inference rules provided here are an adaptation of those given in [16], with additional side conditions to check that the component calculi C_i of logic L_i can derive specific i-formulae.

The resolution rules are applied to a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C}$ in $\mathcal{E}_{conn}$-NF until no new clauses are generated or a contradiction (in the form of $\mathcal{G}_1 \vdash_{\mathsf{C}_1}$ **false**) is found. We assume that our initial problem is a 1-problem, that is, before the $\mathcal{E}_{conn}$-NF procedure, we are given a 1-formula. Then, contradictions found in the second component do not necessarily mean the whole problem is unsatisfiable.

In the following, let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem in $\mathcal{E}_{conn}$-NF; l, l_i be literals; $a \in \mathcal{A}$ and $k \in \mathcal{E}_i$.

The first inference rule, [$\mathcal{E}$-**GEN1**], is a hyperresolution resolution rule [17], in the sense that it resolves multiple clauses at once, while avoiding unnecessary intermediate clauses. Note that in this rule, we may have that $m = 0$. We can only apply the rule if the calculus of the other component, $\mathsf{C}_{\bar{i}}$, can derive at least one of the literals in the scope of modalities on the right-hand side of the implications. This is because, if we have that at least one of the literals in the clause $l_1 \vee \cdots \vee l_m \vee l$ holds, then we could have a world where $\neg l'_1 \wedge \cdots \wedge \neg l'_m \wedge \neg l'$ holds, which means that we cannot have all the literals in the antecedents of the clauses satisfiable at the same time.

$$\begin{array}{rl} \multicolumn{2}{c}{(\text{if } \mathcal{G}_{\bar{i}} \cup \mathcal{K}_{\bar{i}} \vdash_{\mathsf{C}_{\bar{i}}} l_1 \vee \cdots \vee l_m \vee l)} \\ l'_1 \to \boxed{\boxed{k}}^{i} \neg l_1 & \in \mathcal{K}_i \\ \vdots & \\ l'_m \to \boxed{\boxed{k}}^{i} \neg l_m & \in \mathcal{K}_i \\ l' \to \langle\!\langle k \rangle\!\rangle^{i} \neg l & \in \mathcal{K}_i \\ \hline \neg l'_1 \vee \cdots \vee \neg l'_m \vee \neg l' & \in \mathcal{G}_i \end{array} \qquad [\mathcal{E}\textbf{-GEN1}]$$

The second inference rule, [$\mathcal{E}$**-GEN2**], is similar to [$\mathcal{E}$**-GEN1**], but in this case the contradiction occurs between the literals in the scope of modal operators on the right-hand side of the *positive* $\mathcal{E}_i^k$-clauses and a literal clause in the other component language. Note that we may not have that $m = 0$, else the side condition becomes $\mathcal{G}_{\bar{i}} \vdash_{\mathsf{C}_{\bar{i}}} \varnothing$, which means that the empty clause must hold, which contradicts with the fact that the negative modal clause in the premises is satisfiable.

$$\begin{array}{rl} \multicolumn{2}{c}{(\text{if } \mathcal{G}_{\bar{i}} \cup \mathcal{K}_{\bar{i}} \vdash_{\mathsf{C}_{\bar{i}}} l_1 \vee \cdots \vee l_m)} \\ l'_1 \to \boxed{\boxed{k}}^{i} \neg l_1 & \in \mathcal{K}_i \\ \vdots & \\ l'_m \to \boxed{\boxed{k}}^{i} \neg l_m & \in \mathcal{K}_i \\ l' \to \langle\!\langle k \rangle\!\rangle^{i} l & \in \mathcal{K}_i \\ \hline \neg l'_1 \vee \cdots \vee \neg l'_m \vee \neg l' & \in \mathcal{G}_i \end{array} \qquad [\mathcal{E}\textbf{-GEN2}]$$

The inference rules [$\mathcal{E}$**-GEN1**] and [$\mathcal{E}$**-GEN2**] are applied to formulae in a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem and we rely on the decision procedures C_1 and C_2 for the component logics L_1 and L_2 to check the side conditions. This was not necessary in [16], as the calculus was specifically designed to deal with the connections of the logics given there. As we rely only on these decision procedures, we cannot make use of the structure of the proofs produced by these methods. However, the fact that C_1 and C_2 are decision procedures for L_1 and L_2, respectively, is enough to ensure that the calculus presented here is complete with respect to the intended semantics, as shown in Chapter 5.

With these inference rules, we can formally state the definition of our resolution calculus:

Definition 24. Given two decidable logics L_1 and L_2 with their terminating, strongly sound and strongly complete calculi C_1 and C_2, the resolution calculus for the $\mathcal{E}$-connected combination

$\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$, $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$, is the tuple $\langle \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{A} = \varnothing$ is the set of axioms and $\mathcal{R} = \{\mathcal{E}\textbf{-GEN1}, \mathcal{E}\textbf{-GEN2}\}$ is the set of *inference rules*.

Definition 25. Let $\mathbb{C} = \langle \langle \mathcal{G}_1, \mathcal{K}_1 \rangle, \langle \mathcal{G}_2, \mathcal{K}_2 \rangle \rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem into $\mathcal{E}_{conn}$-NF. A *derivation* from $\mathbb{C}$ by $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ is a sequence $(\mathbb{C}_0, \mathbb{C}_1, \mathbb{C}_2, ...)$ of problems such that $\mathbb{C}_0 = \mathbb{C}$, $\mathbb{C}_i = \langle \langle \mathcal{G}_1^i, \mathcal{K}_1^i \rangle, \langle \mathcal{G}_2^i, \mathcal{K}_2^i \rangle \rangle$ and $\mathbb{C}_{i+1}$ is either:

- $\langle \langle \mathcal{G}_1^i \cup \{D\}, \mathcal{K}_1^i \rangle, \langle \mathcal{G}_2^i, \mathcal{K}_2^i \rangle \rangle$, where D is the conclusion of an application of $\mathcal{E}$**-GEN1** or $\mathcal{E}$**-GEN2** to a set of clauses in $\mathcal{K}_1^{i-1}$; or
- $\langle \langle \mathcal{G}_1^i, \mathcal{K}_1^i \rangle, \langle \mathcal{G}_2^i \cup \{D\}, \mathcal{K}_2^i \rangle \rangle$, where D is the conclusion of an application of $\mathcal{E}$**-GEN1** or $\mathcal{E}$**-GEN2** to a set of clauses in $\mathcal{K}_2^{i-1}$

In both cases, D must be in simplified form and not a tautology.

Definition 26. A *refutation* for a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C} = \langle \langle \mathcal{G}_1, \mathcal{K}_1 \rangle, \langle \mathcal{G}_2, \mathcal{K}_2 \rangle \rangle$ by $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ is a finite derivation from $\mathbb{C}$ such that, for some $i \geq 0$, $\mathbb{C}_i = \langle \langle \mathcal{G}_1^i, \mathcal{K}_1^i \rangle, \langle \mathcal{G}_2^i, \mathcal{K}_2^i \rangle \rangle$, we have $\mathcal{G}_1^i \vdash_{\mathsf{C}_1}$ **false**.

Definition 27. We say that a derivation $(\mathbb{C}_0, \mathbb{C}_1, \mathbb{C}_2, ...)$ *terminates* if it is either a refutation or if there exists $i \in \mathbb{N}$ such that any further application of the inference to $\mathbb{C}_i$ produces a clause already in $\mathbb{C}_i$. In this last case, the problem $\mathbb{C}_i$ is called *saturated*.

As the formula to be evaluated for satisfiability is always an 1-formula, we only care for contradictions found in the second component if the first component "sees" it, that is, there is a negative $\mathcal{E}_1$-formula in $\mathcal{K}_1^i$. If this is the case, the contradiction can then be brought to the first component, as becomes clear in the proof of completeness for the calculus, shown in Theorem 6, in Chapter 5.

Example 12. Consider the following 1-formula in NNF:

$$\varphi = \Box^1 (p \lor q) \land \Box^1 \neg p \land \Box^1 \neg q \land \Diamond^1 r$$

where $\{p, q, r\}$ is in $\mathcal{F}(\mathsf{L}_2)$. The corresponding problem in $\mathcal{E}_{conn}$ normal form is:

$$\left\langle \begin{array}{c} \langle \{t_0\}, \{t_0 \to \Box^1 t_1, t_0 \to \Box^1 t_2, t_0 \to \Box^1 t_3, t_0 \to \Diamond^1 t_4\} \rangle, \\ \langle \neg t_1 \lor p \lor q, \neg t_2 \lor \neg p, \neg t_3 \lor \neg q, \neg t_4 \lor r, \varnothing \rangle \end{array} \right\rangle$$

Clauses 1-9 are resulting from the problem above:

$$\begin{array}{lll}
1.\ t_0 & [\mathcal{G}_1] & 6.\ \neg t_3 \vee \neg q \quad [\mathcal{G}_2] \\
2.\ t_0 \rightarrow \boxed{}^1 t_1 & [\mathcal{K}_1] & 7.\ \neg t_2 \vee \neg p \quad [\mathcal{G}_2] \\
3.\ t_0 \rightarrow \boxed{}^1 t_2 & [\mathcal{K}_1] & 8.\ \neg t_4 \vee r \quad [\mathcal{G}_2] \\
4.\ t_0 \rightarrow \boxed{}^1 t_3 & [\mathcal{K}_1] & 9.\ \neg t_1 \vee p \vee q \quad [\mathcal{G}_2] \\
5.\ t_0 \rightarrow \Diamond^1 t_4 & [\mathcal{K}_1] &
\end{array}$$

We may quickly obtain the refutation as follows:

$$10.\ \neg t_0 \quad [\mathcal{E} - \mathbf{GEN2}, 2, 3, 4, 5, \mathcal{G}_2 \vdash_{C_2} \neg t_1 \vee \neg t_2 \vee \neg t_3, \mathcal{G}_1]$$

Then, as $\{t_0, \neg t_0\} \subset \mathcal{G}_1$, and as C_1 is strongly complete, **false** can be derived by C_1, which means that the initial problem is in fact unsatisfiable.

Example 13. In this example, we show the unsatisfiability of a formula on the $\mathcal{E}$-connection of a reflexive logic (T) with a transitive logic ($\mathsf{4}$). In order to improve readability, we will not rename propositional symbols inside non-connecting modalities. This does not affect the proof, as our rules do not directly operate with the modalities in the component logics in any way. We assume sound and complete calculi C_T and C_4 for T and $\mathsf{4}$, respectively.

Consider the following 1-formula of $\mathcal{C}^{\mathcal{M}}(\mathsf{T}, \mathsf{4})$:

$$\varphi = (p \rightarrow \Diamond\Diamond\Diamond\neg(\boxed{b}q \rightarrow \boxed{b}\,\boxed{b}q)) \wedge \boxed{a}p)$$

where $\{p\}$ is in $\mathcal{F}(\mathsf{T})$ and $\{q\}$ is in $\mathcal{F}(\mathsf{4})$. The corresponding problem in $\mathcal{E}_{conn}$ normal form is:

$$\left\langle \begin{array}{c} \langle\{t_0, t_5 \vee \neg t_0 \vee \neg p, \}, \{t_0 \rightarrow \boxed{a}p, t_5 \rightarrow \Diamond^1 t_4, t_3 \rightarrow \Diamond^1 t_2\}\rangle \\ \langle\varnothing, \{t_1 \rightarrow \langle b\rangle\neg q, t_2 \rightarrow \langle b\rangle t_1, t_1 \rightarrow \boxed{b}q, t_4 \rightarrow \Diamond^2 t_3\}\rangle \end{array} \right\rangle$$

Clauses 1-9 are resulting from the problem above:

$$\begin{array}{llll}
1.\ t_0 & [\mathcal{G}_1] & & \\
2.\ t_5 \vee \neg t_0 \vee \neg p & [\mathcal{G}_1] & 6.\ t_1 \rightarrow \langle b\rangle\neg q & [\mathcal{K}_2] \\
3.\ t_0 \rightarrow \boxed{a}p & [\mathcal{K}_1] & 7.\ t_2 \rightarrow \langle b\rangle t_1 & [\mathcal{K}_2] \\
4.\ t_5 \rightarrow \Diamond^1 t_4 & [\mathcal{K}_1] & 8.\ t_1 \rightarrow \boxed{b}q & [\mathcal{K}_2] \\
5.\ t_3 \rightarrow \Diamond^1 t_2 & [\mathcal{K}_1] & 9.\ t_4 \rightarrow \Diamond^2 t_3 & [\mathcal{K}_2]
\end{array}$$

As the second component is $\mathsf{4}$, a transitive logic, note that $\mathcal{G}_2 \vdash_{\mathsf{C}_4} \neg t_2$ must hold. Also, as the first component T is a reflexive logic, if $\neg t_5$ holds, we have $\mathcal{G}_1 \vdash_{\mathsf{C}_\mathsf{T}} \neg t_0$. Therefore, we can obtain a refutation for the problem as follows.

$$\begin{aligned}
&10.\ \neg t_3 \quad [\mathcal{E} - \mathbf{GEN1}, 5, \mathcal{G}_2 \vdash_{C_4} \neg t_2, \mathcal{G}_1]\\
&11.\ \neg t_4 \quad [\mathcal{E} - \mathbf{GEN1}, 9, \mathcal{G}_1 \vdash_{C_T} \neg t_3, \mathcal{G}_2]\\
&12.\ \neg t_5 \quad [\mathcal{E} - \mathbf{GEN1}, 4, \mathcal{G}_2 \vdash_{C_4} \neg t_4, \mathcal{G}_1]
\end{aligned}$$

Then, as $\{t_0, \neg t_0\} \subset \mathcal{G}_1$, and as $\mathsf{C_T}$ is strongly complete, **false** can be derived by $\mathsf{C_T}$, which means that the initial problem is unsatisfiable.

4.3 Algorithm

In this section, we present an algorithm for applying the rules of the calculus. First, we introduce a non-deterministic version of this algorithm, and later we will show the necessary steps needed for our implementation.

```
CONNECTED-RESOLUTION($\mathbb{C}$)
1  do
2      Choose a set of modal candidates, together with a literal clause
       $l_1 \vee ... \vee l_n$ for querying.
3      Apply $\mathcal{E}$-GEN1
4      Apply $\mathcal{E}$-GEN2
5  until a contradiction is found or no new clauses are generated
6  return
```

This version of the algorithm is straightfoward: we non-deterministically choose a set of modal clauses from $\mathbb{C}$ in which we will try to apply $\mathcal{E}$-**GEN1** and $\mathcal{E}$-**GEN2**. Before applying the rules, the side condition of the $\mathcal{E}$-**GEN** rules require that we check if the clause $l_1 \vee ... \vee l_n$ can be derived by the appropriate component. If it can, we add the resolvents to their appropriate sets (if they are new) and check if either the resolvents were already generated or a contradiction can be derived, stopping the execution in these cases. Otherwise, we repeat the whole process, emulating a derivation of the calculus.

For a proper implementation of this algorithm, we need a deterministic method for choosing the modal candidates and the queried clause. First note that, if we always create a *new* literal while renaming subformulae inside modal connecting operators, each modal clause of the form $l_1 \to \boxed{\square}\, l'_1$ is uniquely determined by l'_1. This allow us to generate the set of modal candidates, which will be used on the application of a rule, by the literal clause that will be queried.

With this in mind, we may focus on the generation of the literal clause. Instead of non-deterministically choosing *some* clause φ, we may instead try *all* possible literal clauses for a component C_i in each step, covering any possible choice of a clause.

A simulation of the execution of this deterministic approach is shown in the next example. For simplicity, we focus only on the application of $\mathcal{E}$-**GEN1**.

Example 14. Fig. 4.1 shows the execution of our deterministic algorithm for the problem $\mathbb{C} = \langle\langle l_1 \to \Diamond l_1', l_2 \to \Diamond l_2', l_1\rangle, \langle l_1' \to \Diamond l_2, l_2' \vee \neg p, p\rangle\rangle$. In this example, we choose the query clause by lexicographic order.

We start in component 1 by choosing $\neg l_1'$ for querying, and $\{l_1 \to \Diamond l_1'\}$ as the set of modal candidates. We ask the prover of component 2 if $\neg l_1'$ is derivable, which yields a negative answer (1). We move on to the next attempt.

We then choose $\neg l_2'$ and $\{l_2 \to \Diamond l_2'\}$. We ask the prover of component 2 if $\neg l_2'$ is derivable, which yields a *positive* answer (2). The resolvent of the application of $\mathcal{E}$-**GEN1** is added to component 1 (3).

The $\mathcal{E}$-prover then switches to the other component and chooses the set $\{l_1' \to \Diamond l_2\}$ as the set of modal candidates and the literal clause $\neg l_2$ for the querying. We ask component 1 if $\neg l_2$ is derivable, which yields a positive answer (4). The resolvent $\neg l_1'$ is added to component 2 (5). This ends all possible choices for component 2.

Finally, we choose $\neg l_1'$ for querying, and $\{l_1 \to \Diamond l_1'\}$ as the set of modal candidates. We ask the prover of component 2 if $\neg l_1'$ is derivable, which yields a *positive* answer (6), adding the resolvent $\neg l_1$ to the known clauses of component 1 (7). This, combined with l_1 generates a contradiction on the first component (8), which ends the proof.

4.4 Implementation

A proof-of-concept implementation of this algorithm, currently supporting logics $\mathsf{K}_{(n)}$, T , $\mathsf{4}$ and $\mathsf{S4}$ as components, is available at `www.cic.unb.br/~nalon/#software` [5]. The implementation is based on the K_SP prover [15], which is an implementation of a modal-layered resolution calculus for the $\mathsf{K}_{(n)}$-logic, described in [14]. The implementation in this work follows the representation given in Fig. 3.2, where there is a $\mathcal{E}$-connected reasoner that queries instances of provers for the component logics. The purpose of this implementation is not to provide a high-performance prover, but rather to provide a simple modular prototype in which we can run a much larger amount of examples than can be done by hand, while also demonstrating the possibility of a future fully-featured prover for $\mathcal{E}$-connected logics. Due to this purpose,

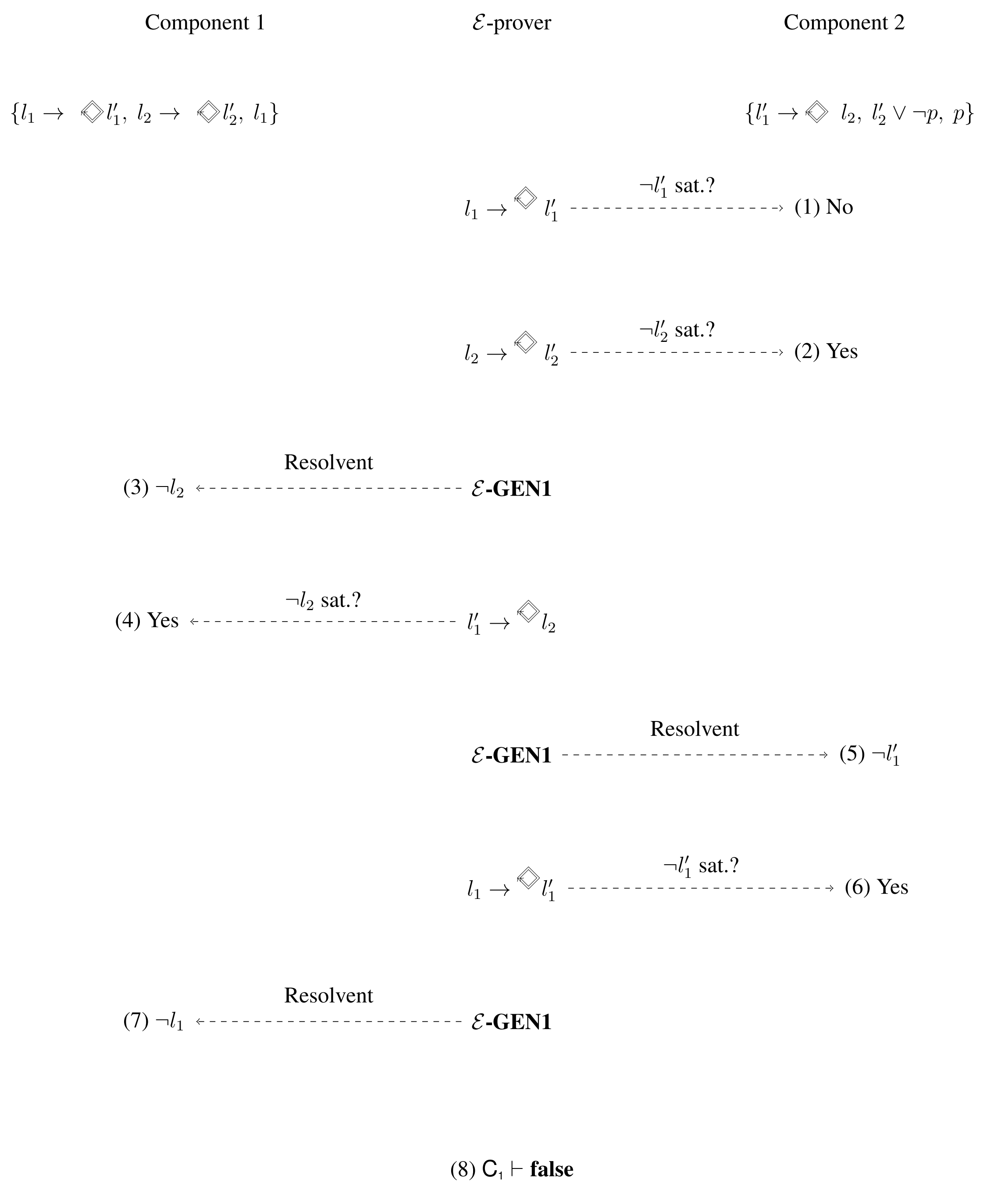


Figure 4.1: Simulation of the querying process

the implementation cannot run on large datasets on a short time, as this was not intended. There are many optimizations to be made, as it will be discussed later.

The KSP prover was chosen both for the proximity of the modal inference rules there implemented with the inference rules given in this work and for the large amount of options it gives us for experimentation, which have been shown necessary to perform adequate preprocessing of the input formulae. Some modifications on the original prover were necessary both to read and store the formulae of each component in an appropriate manner, as well as to correctly perform the resolution of clauses. A new module responsible for managing the communication between the components was also implemented.

For the component provers, we also used *unmodified* KSP instances. This allowed us to get reasonable performance on the components, while also having flexibility on the logics to be connected.

4.4.1 Implementation details

The first modifications to the KSP source code while making the central module were made to the parser and lexer. The new connecting modalities required specific tokens for appropriate representation in the input formulae, and the grammar was also extended to deal with these modalities.

On the next step of the process, we needed to transform the set of clauses to the appropriate $\mathcal{E}_{conn}$ normal form. For this, we slightly modified the original procedure for normal form in KSP, placing each subformula renamed in the structure corresponding to its appropriate set ($\mathcal{G}_1$, $\mathcal{G}_2, \mathcal{K}_1, \mathcal{K}_2$).

Representation of the sets of literal and modal clauses of each component was done using the same chain of hash tables already used by the KSP prover for storing clauses. Each $\mathcal{G}_i$ and $\mathcal{K}_i$-sets are represented by a separate structure. This allows for reuse of the many functions already available for processing the formulae in KSP.

The main processing loop of the program closely follows the algorithm described in Section 4.3. The fact that we ask the satisfiability of all query-clauses multiple times may be attenuated by the use of a technique called *subsumption*.

Subsumption of a clause φ by a clause ψ works by comparing their literals $\mathcal{L}it(\varphi)$ and $\mathcal{L}it(\psi)$: if $\mathcal{L}it(\psi) \subseteq \mathcal{L}it(\varphi)$, then, if ψ is satisfiable, so must be φ, by the definition of disjunction. With this in mind, we may store the clauses which we *know* that can be derived by component $\mathsf{C_i}$ from a set $\mathcal{S}_i$ of clauses, allowing us to check if φ is subsumed by any clause in $\mathcal{S}_i$ and avoiding a test for the derivability of the subsumed clause by the component, which is costly. The set of known clauses is implemented using the same hash-structure used by the $\mathcal{G}_i$-sets, so that we can reuse the subsumption functions of KSP.

In each iteration, we generate a new clause that will be tested for derivability in the respective component, checking for subsumption beforehand. If the clause is subsumed, we do not need to apply the $\mathcal{E}$-**GEN** rules again, as the resolvent would already be subsumed by previous ones. The query-clauses that are not derivable are stored in a list for each component, both for simplicity of implementation and the cyclic approach to querying.

Both rules of the calculus are also implemented reusing existing functions of K$_S$P, with changes to support parametrization of source sets. As per the described algorithm, the program keeps generating new clauses by querying and applying rules until either a contradiction is found or no new clauses are added.

For handling the querying, a new module was added to the program. This module provides a function that receives only the clause to be queried and the component index, handling all comunication. Due to the prototypical nature of this implementation, a simple approach to communication was taken: for each query, an instance of the component prover is initialized, receives all the clauses of the component together with the negated query and returns either satisfiable/unsatisfiable, allowing us to gain information on the derivability of the clause from the set of formulae of the component.

4.4.2 Experiments

One of the challanges presented during this work was the lack of dedicated bases for experimentation. As this work presents, to the best of our knowledge, the first modular calculus and prover for $\mathcal{E}$-connected logics, there are no available collection of test cases and to check the robustness of implementation.

Manual tests were developed to verify correct functioning of this implementation, which cover varying logics for the components. For more extensive tests, we have modified both the test set provided with K$_S$P source code, which was used mostly for verifying correctness, as well as the **LWB** [1] knowledge base, typically used for benchmarking modal logics, which was mostly used for workload testing. These modifications were straightfoward, replacing the basic $\mathsf{K}_{(n)}$ modalities with connecting modalities.

As the purpose of this implementation never was perfomance, running times can be vastly improved. The database which K$_S$P provides consists of small problems and the expected results can be manually verified — with all outputs matching expectations — but provide no meaningful running time. The **LWB** database, however, has problems that are far too large to non-automatically check for satisfiability. Moreover, most of the problems are too large for our current implementation to execute in the determined timeout of 20 minutes. Of the 378 problems in **LWB**, only 60 could run within the configured timeout. These problems are divided in 9 families: `branch`, `d4`, `dum`, `grz`, `lin`, `path`, `ph`, `poly` and `t4p`. Each of these families consists of 21 *satisfiable* problems and 21 *unsatisfiable* problems for $\mathsf{K}_{(n)}$. An overview of the executions for each family is shown in Fig. 4.2, Table 4.3 and Table 4.4. Detailed results are available at the appendix.

Family	Total time (s)	No. of instances solved/total	Average time (s)
branch	7.54	1/21	7.54
d4	179.98	8/21	22.50
dum	0.00	0/21	–
grz	211.93	6/21	35.32
lin	8.34	2/21	4.17
path	87.40	1/21	87.40
ph	0.00	0/21	–
poly	437.51	2/21	218.76
t4p	171.98	9/21	19.11

Table 4.3: Solved satisfiable problems by family

Family	Total time (s)	No. of instances solved/total	Average time (s)
branch	2.34	1/21	2.34
d4	0.00	0/21	–
dum	0.00	0/21	–
grz	0.00	0/21	–
lin	0.01	21/21	0.00
path	54.38	2/21	27.19
ph	51.20	3/21	17.07
poly	1.26	1/21	1.26
t4p	0.00	0/21	–

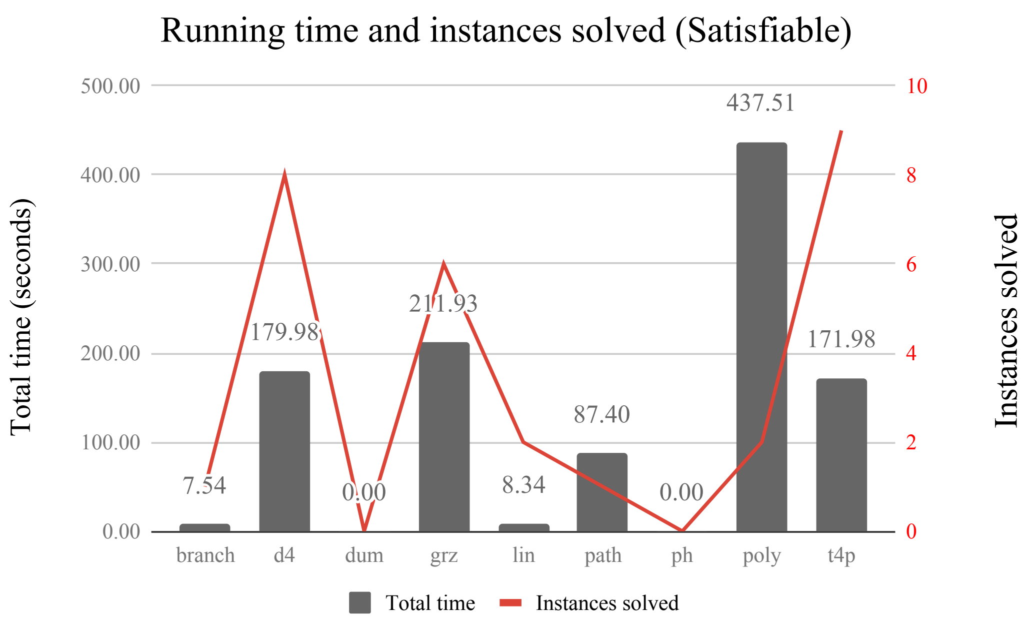
Table 4.4: Solved unsatisfiable problems by family

4.4.3 Possible optimizations

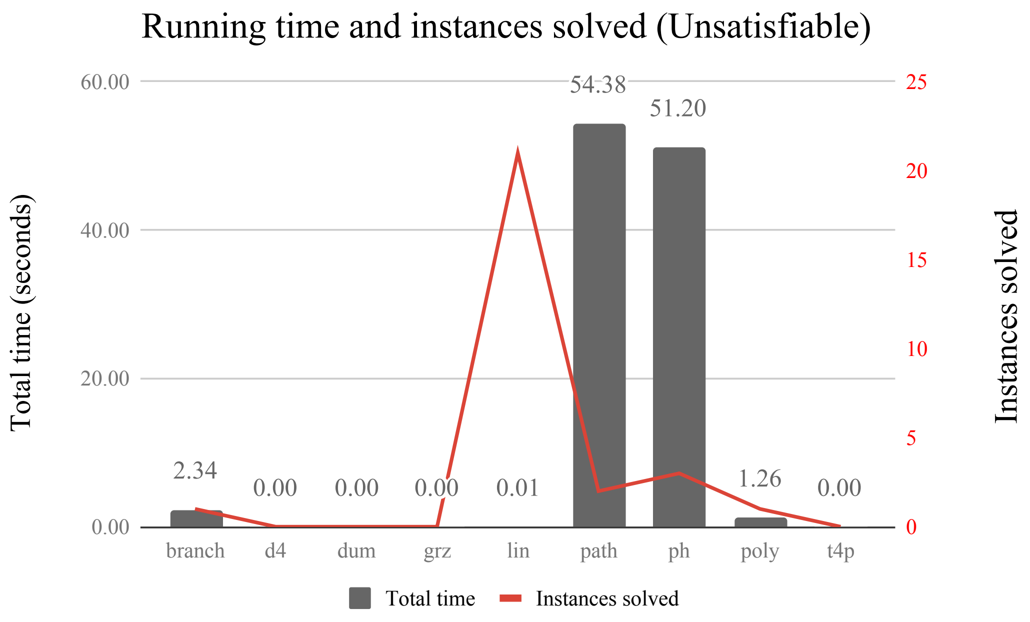
The purpose of this version of the software was merely to provide an evidence of the feasability of an implementation, and many of the functionalities were done in the most straightfoward and simplistical way possible. Therefore, some modifications can highly improve execution times.

Most of the execution time on large exemples is spent on the processing of queries by the component provers. As detailed in Section 4.4.1, for each query, we start a fresh instance and send the component its whole clause set, together with the query-clause itself. Some modifications on the component K$_S$P could allow us to make successive queries while storing the saturated initial clause set in an *usable* set. That way, the component does not need to saturate the whole set each time, greatly improving performance. Also, a fully modular version of this prover might define a simpler protocol for comunication, allowing higher interchangeability of the components.

Our approach to generating queries is also not optimal, as we build literal clauses from *all* the renamed literals. Although we skip clauses that have two or more negative modal literals,



(a) Satisfiable examples



(b) Unsatisfiable examples

Figure 4.2: Total running time of test groups

we still have to do the process of checking. If we split the negative and positive renamed literals in two sets and generate the queries by combining them, we may reduce the amount of times we pass through the loop from 2^{n+p} to $n * 2^p$, where n is the amount of negative modal literals and p is the amount of positive modal literals.

Finally, we might also try to reduce the amount of literals generated in the transformation of the problem to $\mathcal{E}_{conn}$-normal form by reusing literals for equal subformulae. It is important to note, however, that special care must be taken when combining this improved renaming with the previous optimization and the check for subsumption on the algorithm. If a renamed literal appears both in the scope of positive and negative connection modalities, it must be present both on the set of negatives and positive literals. Moreover, even with this replication we might face another problem on the check for subsumption, as a literal no longer uniquely identify a modal clause. A proper approach to renaming might require some deeper study, but the amount of literals added during the transformation has a significant impact on the size of the problem, and any optimization in this regard might be worth the effort.

Chapter 5

Metatheoretical results

This chapter contains the metatheoretical results regarding the transformation of formulae into normal form and the results regarding the calculus itself, including proofs of soundness, completeness and termination.

5.1 Transformation

In this section we show that the transformation of a formula into $\mathcal{E}_{conn}$ normal form is correct and terminating. By correctness, we mean that a formula is satisfiable if, and only if, its transformation into the normal form is satisfiable. By termination, we mean that the transformation process does not go on indefinitely, that is, there is we always obtain a problem $\mathbb{C}'$ such that no transformation rule can be applied. We show that this problem $\mathbb{C}'$ is in normal form.

5.1.1 Correctness of the transformation

Before starting to prove that the transformation of a formula into the normal form preserves satisfiability, we need some auxiliary lemmas that show that satisfiability of a problem only depends on its propositional symbols. This is done in Lemmas 1 to 3.

Lemma 1. *Let φ be an i-formula and t a propositional symbol not occurring in φ. Let $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, where $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$. Let $\mathbb{M}' = \langle \mathbb{W}'_1, \mathbb{W}'_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, where $\mathbb{W}'_{\bar{i}} = \mathbb{W}_{\bar{i}}$ and $\mathbb{W}'_i = \langle \mathcal{W}^i, \pi'_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_m\rangle$, such that, for all $w \in \mathbb{W}_i$, $\pi'_i(w)(p) = \pi_i(w)(p),\ for\ all\ p \neq t$. Let w be a world in $\mathbb{W}_i$. Then, $(\mathbb{M}, w) \models_\ell \varphi$ if, and only if, $(\mathbb{M}', w) \models_\ell \varphi$.*

Proof. By induction on the structure of φ.

We start with the induction basis, which is the case where φ is a propositional symbol. If φ is a propositional symbol p, then by the statement of the lemma we have that $p \neq t$. Then,

$\pi'_i(w)(p) = \pi_i(w)(p)$. This means that p is satisfiable in exactly the same worlds in both models. Then, $(\mathbb{M}, w) \models_\ell p$ if, and only if, $(\mathbb{M}', w) \models_\ell p$.

Now for the other cases we use the inductive hypothesis, which states that, for any world w, for any formula ψ with size smaller than the size of φ, $(\mathbb{M}, w) \models_\ell \psi$ if, and only if, $(\mathbb{M}', w) \models_\ell \psi$.

1. If φ is of the form $\varphi_1 \wedge \varphi_2$, then, by the definition of satisfiability of the conjunction, if $(\mathbb{M}, w) \models_\ell \varphi_1 \wedge \varphi_2$, then $(\mathbb{M}, w) \models_\ell \varphi_1$. Similarly, $(\mathbb{M}, w) \models_\ell \varphi_2$. By the induction hypothesis, as φ_1 and φ_2 are smaller than φ, $(\mathbb{M}', w) \models_\ell \varphi_1$ and $(\mathbb{M}', w) \models_\ell \varphi_2$. Then, by the semantics of conjunction, $(\mathbb{M}', w) \models_\ell \varphi_1 \wedge \varphi_2$. The "only if" part is analogous.

2. If φ is of the form $\varphi_1 \vee \varphi_2$, then, by the definition of satisfiability of the disjunction, if $(\mathbb{M}, w) \models_\ell \varphi_1 \vee \varphi_2$, then either $(\mathbb{M}, w) \models_\ell \varphi_1$ or $(\mathbb{M}, w) \models_\ell \varphi_2$. Consider $(\mathbb{M}, w) \models_\ell \varphi_1$, then, as φ_1 is smaller than φ, by induction hypothesis, $(\mathbb{M}', w) \models_\ell \varphi_1$. Then, by the semantics of the disjunction, $(\mathbb{M}', w) \models_\ell \varphi_1 \vee \varphi_2$. The case for φ_2 is similar. The "only if" part is also analogous.

3. If φ is of the form $\varphi_1 \to \varphi_2$, then, by the definition of satisfiability of the implication, if $(\mathbb{M}, w) \models_\ell \varphi_1$, then $(\mathbb{M}, w) \models_\ell \varphi_2$. Suppose $(\mathbb{M}, w) \models_\ell \varphi_1$. By the semantics of implication, $(\mathbb{M}, w) \models_\ell \varphi_2$. Then, as φ_1 and φ_2 are smaller than φ, by induction hypothesis, $(\mathbb{M}', w) \models_\ell \varphi_1$ and $(\mathbb{M}', w) \models_\ell \varphi_2$. By the semantics of the implication, $(\mathbb{M}', w) \models_\ell \varphi_1 \to \varphi_2$. If $(\mathbb{M}, w) \not\models_\ell \varphi_1$, then, by induction hypothesis, $(\mathbb{M}', w) \not\models_\ell \varphi_1$. Then, by the semantics of implication, $(\mathbb{M}', w) \models_\ell \varphi_1 \to \varphi_2$.

4. If φ is of the form $\neg\varphi_1$, by the semantics of negation, $(\mathbb{M}, w) \not\models_\ell \varphi_1$. Then, as φ_1 is smaller than φ, by induction hypothesis, $(\mathbb{M}', w) \not\models_\ell \varphi_1$. By the semantics of negation, $(\mathbb{M}', w) \models_\ell \neg\varphi_1$.

5. If $\varphi = \boxed{a}\varphi_1$, then by the semantics of $\boxed{a}$, for every world $w' \in \mathbb{W}_i$ such that $w\mathcal{R}^i_a w'$, $(\mathbb{W}_i, w') \models_\ell \varphi_1$. By the induction hypothesis, as φ_1 is smaller than φ, $(\mathbb{W}'_i, w') \models_\ell \varphi_1$. Then, because the relation $\mathcal{R}^i_a$ is the same in both models, by the semantics of $\boxed{a}$, $(\mathbb{W}'_i, w) \models_\ell \boxed{a}\varphi$.

6. If $\varphi = \boxed{j}\varphi_1$, then for every world $w' \in \mathbb{W}_{\bar{i}}$ such that $w\mathcal{E}^i_j w'$, $(\mathbb{W}_{\bar{i}}, w') \models_\ell \varphi_1$. By induction hypothesis, as φ_1 is smaller than φ, $(\mathbb{W}'_{\bar{i}}, w') \models_\ell \varphi_1$ Then, because the relation $\mathcal{E}^i_j$ is the same in both models, by the semantics of $\boxed{j}$, $(\mathbb{M}', w) \models_\ell \boxed{a}\varphi_1$.

7. If $\varphi = \langle a \rangle\varphi_1$, then by the semantics of $\langle a \rangle$, there is a world $w' \in \mathbb{W}_i$ such that $w\mathcal{R}_a w'$ and $(\mathbb{W}_i, w') \models_\ell \varphi_1$. By the induction hypothesis, as φ_1 is smaller than φ, $(\mathbb{W}'_i, w') \models_\ell \varphi_1$. Then, because the relation $\mathcal{R}^i_a$ is the same in both models,by the semantics of $\langle a \rangle$, $(\mathbb{W}'_i, w) \models_\ell \langle a \rangle\varphi_1$.

8. If $\varphi = \Diamond_{j}\varphi_1$, then there is a world $w' \in \mathbb{W}_{\bar{i}}$ such that $w\mathcal{E}^i_j w'$ and $(\mathbb{W}_{\bar{i}}, w') \models_\ell \varphi_1$. By induction hypothesis, as φ_1 is smaller than φ, $(\mathbb{W}'_{\bar{i}}, w') \models_\ell \varphi_1$ Then, because the relation $\mathcal{E}^i_j$ is the same in both models, by the semantics of $\Diamond_{j}$, $(\mathbb{M}', w) \models_\ell \Diamond_{j}\varphi_1$.

As $(\mathbb{M}, w) \models_\ell \varphi$ if, and only if, $(\mathbb{M}', w) \models_\ell \varphi$ holds for every possible form of φ, the lemma holds.

□

Lemma 2. *Let φ be an i-formula and t a propositional symbol not occurring in φ. Let $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, where $\mathbb{W}_i = \langle \mathcal{W}^i, \pi^i, \mathcal{R}^i_1, \dots, \mathcal{R}^i_{n_i}\rangle$. Let $\mathbb{M}' = \langle \mathbb{W}'_1, \mathbb{W}'_2,$ $(\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, where $\mathbb{W}'_{\bar{i}} = \mathbb{W}_{\bar{i}}$ and $\mathbb{W}'_i = \langle \mathcal{W}^i, \pi'_i, \mathcal{R}^i_1, \dots, \mathcal{R}^i_{n_i}\rangle$, such that, for all $w \in \mathbb{W}_i$, $\pi'_i(w)(p) = \pi_i(w)(p),\ for\ all\ p \neq t$. Then, $\mathbb{M} \models_g \varphi$ if, and only if, $\mathbb{M}' \models_g \varphi$.*

Proof. Suppose $\mathbb{M} \models_g \varphi$. This means that, for every world $w \in \mathbb{W}_i$, $(\mathbb{M}, w) \models_\ell \varphi$. By the previous lemma, this means that for every world $w \in \mathbb{W}'_i$, we have that $(\mathbb{M}', w) \models_\ell \varphi$. That is, $\mathbb{M}' \models_g \varphi$. The proof is analogous for the "only if" part. □

Lemma 3. *Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem and t a propositional symbol not occurring in φ. Let $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, where $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \dots,$ $\mathcal{R}^i_{n_i}\rangle$ and $\mathbb{M} \models_g \mathbb{C}$. Let $\mathbb{M}' = \langle \mathbb{W}'_1, \mathbb{W}'_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, where $\mathbb{W}'_{\bar{i}} = \mathbb{W}_{\bar{i}}$ and $\mathbb{W}'_i =$ $\langle \mathcal{W}^i, \pi'_i, \mathcal{R}^i_1, \dots, \mathcal{R}^i_{n_i}\rangle$, such that $\pi'_i(p) = \pi_i(p),\ for\ all\ p \neq t$. Then, $\mathbb{M}' \models_g \mathbb{C}$.*

Proof. Straightforward, as it suffices to apply the previous lemma to each formula in the sets.

□

This concludes the proof that the satisfiability of a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem only depends on the valuation to the propositional symbols occurring in the problem. Next, we show the correctness of each transformation rule.

Lemma 4. *(τ_{init}) Let φ be an i-formula into NNF, and let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $\mathcal{G}_1 = \{t_0, t_0 \to \varphi\}$ and $\mathcal{K}_1 = \mathcal{G}_2 = \mathcal{K}_2 = \varnothing$, where t_0 does not occur in φ. Then, φ is globally satisfiable if and only if $\mathbb{C}$ is globally satisfiable.*

Proof. ($\Rightarrow$) If φ is globally satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_1 = \langle \mathcal{W}^1, \pi_1, \mathcal{R}^1_1, \dots, \mathcal{R}^1_{n_1}\rangle$, such that $\mathbb{M} \models_g \varphi$. We construct a model $\mathbb{M}'$ = $\langle \mathbb{W}'_1, \mathbb{W}'_2,$ $(\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}'_2 = \mathbb{W}_2$, and $\mathbb{W}'_1 = \langle \mathcal{W}^i, \pi'_i, \mathcal{R}^i_1, \dots, \mathcal{R}^i_{n_1}\rangle$, where $\pi'_i(w)(p) =$ $\pi_1(w)(p)$, for all $p \neq t_0$, and $\pi'_i(w)(t_0) = true$, for all $w \in \mathbb{W}_1$, where t_0 is a new propositional symbol not occurring in φ. By Lemma 1, we have that $\mathbb{M}' \models_g \varphi$. As $\mathbb{W}'_1 \models_g \varphi$, by the semantics of implication we have that $\mathbb{W}'_1 \models_g t_0 \to \varphi$. Then we have that $\mathbb{M}' \models_\ell t_0 \to \varphi$. As, by construction, $\pi'_i(w)(t_0) = true$, for all $w \in \mathbb{W}'_1$, we have that $\mathbb{W}'_1 \models_g t_0$ and $\mathbb{M}' \models_g t_0$, and finally $\mathbb{M}' \models_g \{t_0 \to \varphi, t_0\}$, that is, $\mathbb{M}' \models_g \mathcal{K}_1$. Then, by the definition of satisfiability of $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problems, because $\mathcal{G}_2, \mathcal{K}_1, \mathcal{K}_2$ are trivially satisfied in any model, we have that $\mathbb{M}' \models_g \mathbb{C}$.

($\Leftarrow$) If $\mathbb{C}$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j \in \mathcal{I}_1}, (\mathcal{E}^2_k)_{k \in \mathcal{I}_2} \rangle$, with $\mathbb{W}_1 = \langle \mathcal{W}^1, \pi_1, \mathcal{R}^1_1, \ldots, \mathcal{R}^1_{n_1} \rangle$, such that $\mathbb{W}_1 \models_g (t_0 \to \varphi)$ and $\mathbb{W}_1 \models_g t_0$. By the semantics of implication, we have $\mathbb{W}_1 \models_g \varphi$. Finally, we have $\mathbb{M} \models_g \varphi$.

□

This lemma shows that a formula φ is satisfied if, and only if, it is the corresponding initial $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem is satisfiable. The following lemmas deal with the other transformation rules.

Lemma 5. *($\tau_\wedge$) Let φ_1, φ_2 be i-formulae and let t be a literal. Let $\mathbb{C} = \langle \langle \mathcal{G}_1, \mathcal{K}_1 \rangle, \langle \mathcal{G}_2, \mathcal{K}_2 \rangle \rangle$, where $\{t \to (\varphi_2 \wedge \varphi_2)\} \subseteq \mathcal{G}_i$. Then, $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i \rangle$ is satisfiable if, and only if, $\mathcal{C}'_i = \langle \mathcal{G}'_i, \mathcal{K}'_i \rangle$ is satisfiable, where $\mathcal{G}'_i = \{t \to \varphi_1, t \to \varphi_2\} \cup \mathcal{G}_i \setminus \{t \to (\varphi_2 \wedge \varphi_2)\}$ and $\mathcal{K}'_i = \mathcal{K}_i$.*

Proof. ($\Rightarrow$) If $\mathcal{C}_i$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j \in \mathcal{I}_1}, (\mathcal{E}^2_k)_{k \in \mathcal{I}_2} \rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i} \rangle$, such that $\mathbb{M} \models_g \mathcal{G}_i \cup \{(t \to \varphi_2 \wedge \varphi_2)\} \cup \mathcal{K}_i$. Then, for any world $w \in \mathbb{W}_i$, we have that $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1 \wedge \varphi_2)$. We consider two cases:

If t holds at w, then, by the semantics of implication, $(\mathbb{W}_i, w) \models_\ell \varphi_1 \wedge \varphi_2$. This means that we have $(\mathbb{W}_i, w) \models_\ell \varphi_1$ and $(\mathbb{W}_i, w) \models_\ell \varphi_2$. By the semantics of implication, we have $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1)$ and $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_2)$. If t does not hold at w, by the semantics of implication we trivially have $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1)$ and $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_2)$.

As in any world w either t holds or it does not hold, this means that for every world w, $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1)$ and $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_2)$, that is, $(\mathbb{W}_i, w) \models_\ell \{t \to \varphi_1, t \to \varphi_2\}$, and then we have $\mathbb{M} \models_g \mathcal{G}'_i$. Then, we have $\mathbb{M} \models_g \mathcal{C}'_i$.

($\Leftarrow$) If $\mathcal{C}'_i$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j \in \mathcal{I}_1}, (\mathcal{E}^2_k)_{k \in \mathcal{I}_2} \rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i} \rangle$, such that $\mathbb{W}_i \models_g \mathcal{C}'_i$. Then, for any world w, we have that $(\mathbb{W}_i, w) \models_\ell \{(t \to \varphi_1), (t \to \varphi_2)\}$. Again, we then consider two cases:

1. If t holds at w, then, by the semantics of conjunction, $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1)$ and $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_2)$. Then, as t holds, we have $(\mathbb{W}_i, w) \models_\ell \varphi_1$ and $(\mathbb{W}_i, w) \models_\ell \varphi_2$, by the semantics of implication. We then have $(\mathbb{W}_i, w) \models_\ell \varphi_1 \wedge \varphi_2$, and by the semantics of implication, we have $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1 \wedge \varphi_2)$

2. If t does not hold at w, by the semantics of implication we trivially have $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1 \wedge \varphi_2)$.

As in any world w either t holds or it does not hold, this means that for every world w, $(\mathbb{W}_i, w) \models_\ell (t \to \varphi_1 \wedge \varphi_2)$, and then we have $\mathbb{M} \models_g (t \to \varphi_1 \wedge \varphi_2)$. Finally, we have $\mathbb{M} \models_g \mathcal{G}_i$. Then, we have $\mathbb{M} \models_g \mathcal{C}_i$.

□

Lemma 6. *($\tau_\vee$) Let φ_1, φ_2 be i-formulae and let t be a literal. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\{t \to \varphi_1 \vee \varphi_2\} \subseteq \mathcal{G}_i$. Then, $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ is satisfiable if, and only if, $\mathcal{C}'_i = \langle \mathcal{G}'_i, \mathcal{K}'_i\rangle$ is satisfiable, where t_1 is a new propositional symbol and $\mathcal{G}'_i = \{t \to \varphi_1 \vee t_1, t_1 \to \varphi_2\} \cup \mathcal{G}_i \setminus \{t \to \varphi_1 \vee \varphi_2\}$, $\mathcal{K}'_i = \mathcal{K}_i$.*

Proof. ($\Rightarrow$) If $\mathcal{C}_i$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j \in \mathcal{I}_1}, (\mathcal{E}^2_k)_{k \in \mathcal{I}_2}\rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$, such that $\mathbb{M} \models_g \mathcal{C}_i$. We construct a model $\mathbb{M}' = \langle \mathbb{W}'_1, \mathbb{W}'_2, (\mathcal{E}^1_j)_{j \in \mathcal{I}_1}, (\mathcal{E}^2_k)_{k \in \mathcal{I}_2}\rangle$, with $\mathbb{W}'_{\bar{i}} = \mathbb{W}_{\bar{i}}$, and $\mathbb{W}'_i = \langle \mathcal{W}^i, \pi'_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$, where $\pi'_i(w)(p) = \pi_i(w)(p)$, for all $p \neq t_1$, and $\pi'_i(w)(t_1) = true$, if, and only if, $(\mathbb{W}_i, w) \models_\ell \varphi_2$. By Lemma 3, $\mathcal{C}_i$ is satisfiable. Then, by construction of our model, $t_1 \to \varphi_2$ is trivially satisfied, as t_1 holds everywhere φ_2 holds. Now, if we have that $(t \to \varphi_1 \vee \varphi_2)$ is satisfied at a world w, then if t holds, either φ_1 or φ_2 is satisfied at w. As we have that $\pi'_i(w)(t_1) = true$ if and only if φ_2 holds at w, then, by the semantics of disjunction, $\varphi_1 \vee t_1$ is satisfied and then $t \to \varphi_1 \vee t_1$ is satisfied. By the semantics of conjunction, $\{(t \to \varphi_1 \vee t_1), (t_1 \to \varphi_2)\}$ is satisfied. If t does not hold at w, then the whole formula is also trivially satisfied. Then, as this holds at every world of $\mathbb{W}'_i$, we have that $\mathbb{M}' \models_g \{(t \to \varphi_1 \vee t_1), (t_1 \to \varphi_2)\}$, that is, $\mathbb{M}' \models_g \mathcal{C}'_i$.

($\Leftarrow$) If $\mathcal{C}'_i$ is satisfiable, both $(t \to \varphi_1 \vee t_1)$ and $(t_1 \to \varphi_2)$ are satisfied at every world w. Then, for any world w, if t does not hold at w, $(t \to \varphi_1 \vee \varphi_2)$ is trivially satisfied, by the semantics of implication. If t does hold at w, then, by the semantics of implication and disjunction, either φ_1 or t_1 holds at w. If φ_1 holds at w, then, by the semantics of the disjunction, $\varphi_1 \vee \varphi_2$ holds and then $t \to \varphi_1 \vee \varphi_2$ holds. If t_1 holds, as $t_1 \to \varphi_2$ holds at w, by the semantics of the implication, φ_2 holds. Then, we have that $\varphi_1 \vee \varphi_2$ must hold, by the semantics of disjunction, and then we have that $t \to \varphi_1 \vee \varphi_2$ holds by the semantics of implication. Finally, $\mathcal{C}_i$ is satisfied.

□

Lemma 7. *($\tau_\to$) Let φ be an i-formula and let t be a literal. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\{t \to \varphi\} \subseteq \mathcal{G}_i$. If φ is a disjunction of literals, then, $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ is satisfiable if, and only if, $\mathcal{C}'_i = \langle \mathcal{G}'_i, \mathcal{K}'_i\rangle$ is satisfiable and $\mathcal{G}'_i = \{\neg t \vee \varphi\} \cup \mathcal{G}_i \setminus \{t \to \varphi\}$, $\mathcal{K}'_i = \mathcal{K}_i$.*

Proof. ($\Rightarrow$) If $\mathcal{C}_i$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j \in \mathcal{I}_1}, (\mathcal{E}^2_k)_{k \in \mathcal{I}_2}\rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$, such that $\mathbb{M} \models_g \mathcal{G}_i \cup \mathcal{K}_i$. Then, for any world $w \in \mathbb{W}_i$, we have that $(\mathbb{W}_i, w) \models_\ell (t \to \varphi)$. We consider two cases:

If t holds, then, by the semantics of implication, $(\mathbb{W}_i, w) \models_\ell \varphi$. Then, by the semantics of disjunction, $(\mathbb{W}_i, w) \models_\ell \neg t \vee \varphi$. If t does not hold at w, then, by the semantics of negation, $\neg t$ holds, and by the semantics of disjunction, $(\mathbb{W}_i, w) \models_\ell \neg t \vee \varphi$. As $\neg t \vee \varphi$ holds in both cases, $(\mathbb{W}_i, w) \models_\ell \neg t \vee \varphi$ must hold, and as w was arbitrary, then we have $\mathbb{M} \models_g \neg t \vee \varphi$ and $\mathbb{M} \models_g \mathcal{C}'_i$.

($\Leftarrow$) If $\mathcal{C}_i'$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}_j^1)_{j\in\mathcal{I}_1}, (\mathcal{E}_k^2)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}_1^i, \ldots, \mathcal{R}_{n_i}^i\rangle$, such that $\mathbb{W}_i \models_g \mathcal{C}_i'$. Then, for any world w, we have that $(\mathbb{W}_i, w) \models_\ell \neg t \vee \varphi$. Again, we then consider two cases:

If t holds, then, $\neg t$ does not hold and, by the semantics of disjunction, φ holds. Then, by the semantics of implication, $(\mathbb{W}_i, w) \models_\ell t \to \varphi$. If t does not hold, then $t \to \varphi$ holds trivially. Finally, as w was arbitrary, $\mathbb{M} \models_g t \to \varphi$ and $\mathbb{M} \models_g \mathcal{C}_i$. □

Lemma 8. *(τ_{move}) Let φ be an i-formula, and let t be a literal. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\{t \to \varphi\} \subseteq \mathcal{G}_i$. Then, $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ is satisfiable if, and only if, $\mathcal{C}_i' = \langle \mathcal{G}_i', \mathcal{K}_i'\rangle$ is satisfiable, where $\mathcal{G}_i' = \mathcal{G}_i \setminus \{t \to \varphi\}$, $\mathcal{K}_i' = \mathcal{K}_i \cup \{t \to \varphi\}$.*

Proof. Immediate from the definition of satisfiability of a problem, as both $\mathcal{C}_i$ and $\mathcal{C}_i'$ contain the same formulae. □

Lemma 9. *($\tau_{\Box}$) Let φ be an i-formula, and let t be a literal. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\{t \to \boxed{a}\varphi\} \subseteq \mathcal{K}_i$. Then, $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ is satisfiable if, and only if, $\mathcal{C}_i' = \langle \mathcal{G}_i', \mathcal{K}_i'\rangle$ is satisfiable, where t_1 is a new propositional symbol and $\mathcal{K}_i' = \{t \to \boxed{a}t_1\} \cup \mathcal{K}_i \setminus \{t \to \boxed{a}\varphi\}$, $\mathcal{G}_i' = \mathcal{G}_i \cup \{t_1 \to \varphi\}$.*

Proof. ($\Rightarrow$) If $\mathcal{C}_i$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}_j^1)_{j\in\mathcal{I}_1}, (\mathcal{E}_k^2)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}_1^i, \ldots, \mathcal{R}_{n_i}^i\rangle$, such that $\mathbb{M} \models_g \mathcal{C}_i$. We construct a model $\mathbb{M}' = \langle \mathbb{W}_1', \mathbb{W}_2', (\mathcal{E}_j^1)_{j\in\mathcal{I}_1}, (\mathcal{E}_k^2)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_{\bar{i}}' = \mathbb{W}_{\bar{i}}$, and $\mathbb{W}_i' = \langle \mathcal{W}^i, \pi_i', \mathcal{R}_1^i, \ldots, \mathcal{R}_{n_i}^i\rangle$, where $\pi_i'(w)(p) = \pi_i(w)(p)$, for all $p \neq t_1$, and $\pi_i'(w)(t_1) = true$, if, and only if, $(\mathbb{W}_i, w) \models_\ell \varphi$. By Lemma 3, $\mathbb{M}' \models_g \mathcal{C}_i$. By construction of $\mathbb{M}'$, $t_1 \to \varphi$ is trivially satisfied, as t_1 holds everywhere φ holds. Now, if we have that $(t \to \boxed{a}\varphi)$ is satisfied at a world $w \in \mathbb{W}_i$, then if t holds, by the semantics of implication and $\boxed{a}$, for every world $w' \in \mathbb{W}_i$, such that $w\mathcal{R}_a w'$, we have that $(\mathbb{W}_i, w') \models_\ell \varphi$. As we have that $\pi_i'(w)(t_1) = true$ if and only if φ holds at w, then, by the semantics of $\boxed{a}$, $\boxed{a}t_1$ is satisfied at w and then $t \to \boxed{a}t_1$ is satisfied. By the semantics of conjunction, $(t \to \boxed{a}t_1) \wedge (t_1 \to \varphi)$ is satisfied. If t does not hold at w, then the whole formula is also trivially satisfied. Then, as this is true at every world of $\mathbb{W}_i'$, we have that $\mathbb{M}' \models_g (t \to \boxed{a}t_1) \wedge (t_1 \to \varphi)$. Finally, $\mathcal{C}_i'$ is satisfied.

($\Leftarrow$) If $\mathcal{C}_i'$ is satisfiable, both $(t \to \boxed{a}t_1)$ and $(t_1 \to \varphi)$ are satisfied at every world $w \in \mathbb{W}_i$. Then, for any world w, if t does not hold w, $(t \to \boxed{a}\varphi)$ is trivially satisfied, by the semantics of implication. If t does hold at w, then, by the semantics of implication, $\boxed{a}t_1$ holds at w. By the semantics of $\boxed{a}$, for every world w' such that $w\mathcal{R}_a w'$, we have that t_1 holds at w'. As $t_1 \to \varphi$ holds at every world w, then, by the semantics of implication, we have that for every w', such that $w\mathcal{R}_a w'$, φ holds at w'. Then, by the semantics of $\boxed{a}$, $\boxed{a}\varphi$ holds at w. Thus, by the semantics of implication, $t \to \boxed{a}\varphi$ is satisfiable in $\mathbb{M}$. Finally, $\mathcal{C}_i$ is satisfied.

□

Lemma 10. *($\tau_{\Diamond}$) Let φ be an i-formula and t be a literal. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\{t \to \langle a\rangle\varphi\} \subseteq \mathcal{K}_i$. Then, $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ is satisfiable if, and only if, $\mathcal{C}'_i = \langle \mathcal{G}'_i, \mathcal{K}'_i\rangle$ is satisfiable, where $\mathcal{K}'_i = \{t \to \langle a\rangle t_1\} \cup \mathcal{K}_i \setminus \{t \to \langle a\rangle\varphi\}$, with t_1 a propositional symbol not occurring in $\mathbb{C}$, $\mathcal{G}'_i = \mathcal{G}_i \cup \{t_1 \to \varphi\}$.*

Proof. ($\Rightarrow$) If $\mathcal{C}_i$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$, such that $\mathbb{M} \models_g (t \to \langle a\rangle\varphi)$. We construct a model $\mathbb{M}' = \langle \mathbb{W}'_1, \mathbb{W}'_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}'_{\bar{i}} = \mathbb{W}_{\bar{i}}$, and $\mathbb{W}'_i = \langle \mathcal{W}^i, \pi'_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$, where $\pi'_i(w)(p) = \pi_i(w)(p)$, for all $p \neq t_1$, and $\pi'_i(w)(t_1) = true$, if, and only if, $(\mathbb{W}_i, w) \models_\ell \varphi$. By construction of our model, $t_1 \to \varphi$ is trivially satisfied, as t_1 holds at every world where φ holds. Now, if we have that $(t \to \langle a\rangle\varphi)$ is satisfied at a world $w \in \mathbb{W}_i$, then if t holds, by the semantics of implication and $\langle a\rangle$, there exists a world $w' \in \mathbb{W}_i$, such that $w\mathcal{R}_a w'$, and $(\mathbb{W}_i, w') \models_\ell \varphi$. As we have that $\pi'_i(w)(t_1) = true$ if and only if φ holds at w, then, by the semantics of $\langle a\rangle$, $\langle a\rangle t_1$ is satisfied at w, that is, $\langle a\rangle t_1$ is satisfied, and then $t \to \langle a\rangle t_1$ is satisfied. Then, $\{(t \to \langle a\rangle t_1), (t_1 \to \neg\varphi)\}$ is satisfied. If t does not hold at w, then the whole formula is also trivially satisfied. Then, as this is true at every world of $\mathbb{W}'_i$, we have that $\mathbb{M}' \models_g \{(t \to \langle a\rangle t_1), (t_1 \to \varphi)\}$. Finally, $\mathcal{C}'_i$ is satisfied.

($\Leftarrow$) If $\mathcal{C}'_i$ is satisfiable, both $(t \to \langle a\rangle t_1)$ and $(t_1 \to \varphi)$ are satisfied at every world $w \in \mathbb{W}_i$. Then, for any world w, if t does not hold at w, $(t \to \langle a\rangle\varphi)$ is trivially satisfied, by the semantics of implication. If t does hold at w, then, by the semantics of implication, $\langle a\rangle t_1$ holds at w. By the semantics of $\langle a\rangle$, there exists a world w' such that $w\mathcal{R}_a w'$, and t_1 holds at w'. As $t_1 \to \varphi$ holds at every world w, by the semantics of implication, $\neg\varphi$ holds at w'. That is, for a world w' such that $w\mathcal{R}_a w'$, we have that φ holds at w'. Then, by the semantics of $\langle a\rangle$, $\langle a\rangle\varphi$ holds at w, that is, $\langle a\rangle\varphi$ holds. Thus, by the semantics of implication, $t \to \langle a\rangle\varphi$ is satisfiable in $\mathbb{M}$. Finally, $\mathcal{C}_i$ is satisfied.

□

Lemma 11. *($\tau_{\Box}$) Let φ be an i-formula, and let t be a literal. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\{t \to [k]\varphi\} \subseteq \mathcal{K}_i$. Then, $\mathbb{C}$ is satisfiable if, and only if, $\mathbb{C}' = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ is satisfiable, where t_1 is a new propositional symbol and $\mathcal{K}'_i = \{t \to [k]t_1\} \cup \mathcal{K}_i \setminus \{t \to [k]\varphi\}$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}} \cup \{t_1 \to \varphi\}$, $\mathcal{G}'_i = \mathcal{G}_i$ and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$.*

Proof. First, consider the case where $\mathcal{G}_1 \cup \mathcal{K}_1 \not\models_g \langle k\rangle^1$**true**, for all 1-connecting modalities $\langle k\rangle^1$. This means that the second component is unreachable, and the satisfiability of the problem only depends on $\mathcal{C}_1 = \langle \mathcal{G}_1, \mathcal{K}_1\rangle$. If $i = 2$, the proof is straightforward, as we may extend any model for $\mathbb{C}$ with $\pi_1(w)(t_1) = false$, and thus the application of the rule does not change satisfiability of the problem. If $i = 1$, both $[k]^1\varphi$ and $[k]^1 t_1$ are trivially satisfied, as the second component is unreachable. Then, satisfiability of the problem is also preserved. We may now proceed considering that the models will globally satisfy $\mathcal{G}_2 \cup \mathcal{K}_2$, as we have that $\mathcal{G}_1 \cup \mathcal{K}_1 \models_g \langle k\rangle^1$**true**.

($\Rightarrow$) If $\mathbb{C}$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$ and $\mathbb{W}_{\bar{i}} = \langle \mathcal{W}^{\bar{i}}, \pi_{\bar{i}}, \mathcal{R}^{\bar{i}}_1, \ldots, \mathcal{R}^{\bar{i}}_{n_i}\rangle$, such that $\mathbb{W}_i \models_g \langle \mathcal{G}_i,\ \mathcal{K}_i\rangle$ and $\mathbb{W}_{\bar{i}} \models_g \langle \mathcal{G}_{\bar{i}},\ \mathcal{K}_{\bar{i}}\rangle$. We construct a model $\mathbb{M}' = \langle \mathbb{W}'_1, \mathbb{W}'_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}'_i = \mathbb{W}_i$, and $\mathbb{W}'_{\bar{i}} = \langle \mathcal{W}^{\bar{i}}, \pi'_{\bar{i}}, \mathcal{R}^{\bar{i}}_1, \ldots, \mathcal{R}^{\bar{i}}_{n_i}\rangle$, where $\pi'_{\bar{i}}(w)(p) = \pi_{\bar{i}}(w)(p)$, for all $p \neq t_1$, and $\pi'_{\bar{i}}(w)(t_1) = true$, if, and only if, $(\mathbb{W}_{\bar{i}}, w) \models_\ell \varphi$. By Lemma 3, $\mathbb{M}' \models_g \mathbb{C}$. By construction of our model, $t_1 \to \varphi$ is trivially satisfied at every world $w' \in \mathbb{W}'_{\bar{i}}$, as t_1 holds everywhere φ holds. Then, we have that $\mathbb{M} \models_g t_1 \to \varphi$ and $\mathbb{M} \models_g \langle \mathcal{G}_{\bar{i}}, \mathcal{K}_{\bar{i}}\rangle$.

Now, if we have that $\mathbb{C}'$ is satisfied at a world $w \in \mathbb{W}'_i$, then if t holds, by the semantics of implication and $\boxed{k}$, for every world $w' \in \mathbb{W}_{\bar{i}'}$, such that $w\mathcal{E}^i_k w'$, we have that $(\mathbb{W}'_{\bar{i}}, w') \models_\ell \varphi$. As we have that $\pi'_i(w)(t_1) = true$ if and only if φ holds at w, then, by the semantics of $\boxed{k}$, $\boxed{k}\ t_1$ is satisfied at w and then $t \to \boxed{k}\ t_1$ is satisfied. If t does not hold at w, the implication is also trivially satisfied. Then, we have that $\mathbb{M}' \models_g t \to \boxed{k}\ t_1$ and $\mathbb{M}' \models_g \langle \mathcal{G}_i,\ \mathcal{K}_i\rangle$.

As we have both $\mathbb{M}' \models_g \langle \mathcal{G}_i,\ \mathcal{K}_i\rangle$ and $\mathbb{M}' \models_g \langle \mathcal{G}_{\bar{i}},\ \mathcal{K}_{\bar{i}}\rangle$, then $\mathbb{M}' \models_g \langle\langle \mathcal{G}_i,\ \mathcal{K}_i\rangle, \langle \mathcal{G}_{\bar{i}},\ \mathcal{K}_{\bar{i}}\rangle\rangle$ and the problem $\mathbb{C}'$ is satisfied.

($\Leftarrow$) If $\mathbb{C}'$ is satisfiable, then, by the semantics of conjunction, both $(t \to \boxed{k} t_1)$ and $(t_1 \to \varphi)$ are satisfied at every world w. Then, for any world w, if t does not hold w, $(t \to \boxed{k}\varphi)$ is trivially satisfied, by the semantics of implication. If t does hold at w, then, by the semantics of implication, $\boxed{k}\ t_1$ holds at w. By the semantics of $\boxed{k}$, for every world w' such that $w\mathcal{E}^i_k w'$, we have that t_1 holds at w'. As $t_1 \to \varphi$ holds at every world w, then, by the semantics of implication, we have that for every w' such that $w\mathcal{E}^i_k w'$, φ holds at w'. Then, by the semantics of $\boxed{k}$, $\boxed{k}\ \varphi$ holds at w. Thus, by the semantics of implication, $t \to \boxed{k}\ \varphi$ is satisfiable in $\mathbb{M}$. Finally, $\mathbb{C}$ is satisfied.

$\square$

Lemma 12. *($\tau_{\Diamond}$) Let φ be an i-formula, and let t be a literal. Let $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, where $\{t \to \Diamond_k\varphi\} \subseteq \mathcal{K}_i$. Then, $\mathbb{C}$ is satisfiable if, and only if, $\mathbb{C}' = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ is satisfiable, where $\mathcal{K}'_i = \{t \to \Diamond_k t_1\} \cup \mathcal{K}_i \setminus \{t \to \Diamond_k\varphi\}$, with t_1 a propositional symbol not occurring in $\mathbb{C}$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}} \cup \{t_1 \to \varphi\}$, $\mathcal{G}'_i = \mathcal{G}_i$ and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$.*

Proof. First, consider the case where $\mathcal{G}_1 \cup \mathcal{K}_1 \not\models_g \Diamond_k^1\mathbf{true}$, for all 1-connecting modalities $\Diamond_k^1$. This means that the second component is unreachable, and the satisfiability of the problem only depends on $\mathcal{C}_1 = \langle \mathcal{G}_1, \mathcal{K}_1\rangle$. If $i = 2$, the proof is straightforward, as we may extend any model for $\mathbb{C}$ with $\pi_1(w)(t_1) = false$, and thus the application of the rule does not change satisfiability of the problem. If $i = 1$, both $\Diamond_k^1\varphi$ and $\Diamond_k^1 t_1$ cannot be satisfied, as the second component is unreachable. Then, satisfiability of the problem is also preserved. We may now proceed considering that the models will globally satisfy $\mathcal{G}_2 \cup \mathcal{K}_2$, as we have that $\mathcal{G}_1 \cup \mathcal{K}_1 \models_g \Diamond_k^1\mathbf{true}$.

($\Rightarrow$) If $\mathbb{C}$ is satisfiable, then there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_i = \langle \mathcal{W}^i, \pi_i, \mathcal{R}^i_1, \ldots, \mathcal{R}^i_{n_i}\rangle$, such that $\mathbb{M} \models_g (t \to \Diamond_k\varphi)$. We construct a model $\mathbb{M}' =$

$\langle \mathbb{W}'_1, \mathbb{W}'_2, (\mathcal{E}^1_j)_{j \in \mathcal{I}_1}, (\mathcal{E}^2_k)_{k \in \mathcal{I}_2} \rangle$, with $\mathbb{W}'_i = \mathbb{W}_i$, and $\mathbb{W}'_{\bar{i}} = \langle \mathcal{W}^{\bar{i}}, \pi'_{\bar{i}}, \mathcal{R}^{\bar{i}}_1, \ldots, \mathcal{R}^{\bar{i}}_{n_i} \rangle$, where $\pi'_{\bar{i}}(w)(p) = \pi_{\bar{i}}(w)(p)$, for all $p \neq t_1$, and $\pi'_{\bar{i}}(w)(t_1) = true$, if, and only if, $(\mathbb{W}_{\bar{i}}, w) \models_\ell \varphi$. By construction of our model, $t_1 \to \varphi$ is trivially satisfied, as t_1 holds everywhere φ holds. Then, we have that $\mathbb{M} \models_g \langle \mathcal{G}_{\bar{i}} \cup \{(t_1 \to \varphi)\}, \mathcal{K}_{\bar{i}} \rangle$. Now, if we have that $(t \to \Diamond_k \varphi)$ is satisfied at a world $w \in \mathbb{W}_i$, then if t holds, by the semantics of implication and $\Diamond_k$, there exists a world $w' \in \mathbb{W}_{\bar{i}}$, such that $w \mathcal{E}^i_k w'$, and $(\mathbb{W}_{\bar{i}}, w') \models_\ell \varphi$. As we have that $\pi'_i(w)(t_1) = true$ if and only if φ holds at w, then, by the semantics of $\Diamond_k$, $\Diamond_k t_1$ is satisfied at w, that is, $\Diamond_k t_1$ is satisfied, and then $t \to \Diamond_k t_1$ is satisfied. Then, we have that $\mathbb{M}' \models_g (t \to \Diamond_k\, t_1)$ and $\mathbb{M}' \models_g \langle \mathcal{G}_i, \mathcal{K}_i \rangle$.

As we have both $\mathbb{M}' \models_g \langle \mathcal{G}_i, \mathcal{K}_i \rangle$ and $\mathbb{M}' \models_g \langle \mathcal{G}_{\bar{i}}, \mathcal{K}_{\bar{i}} \rangle$, then $\mathbb{M}' \models_g \langle \langle \mathcal{G}_i, \mathcal{K}_i \rangle, \langle \mathcal{G}_{\bar{i}}, \mathcal{K}_{\bar{i}} \rangle \rangle$ and the problem $\mathbb{C}'$ is satisfied.

($\Leftarrow$) If $\mathbb{C}'$ is satisfiable, $(t \to \Diamond_k t_1)$ is satisfiable at every world in $\mathbb{W}_i$ and $(t_1 \to \varphi)$ is satisfied at every world in $\mathbb{W}_{\bar{i}}$. Then, for any world $w \in \mathbb{W}_i$, if t does not hold at w, $(t \to \Diamond_k \varphi)$ is trivially satisfied, by the semantics of implication. If t does hold at w, then, by the semantics of implication, $\Diamond_k t_1$ holds at w. By the semantics of $\Diamond_k$, there exists a world $w' \in \mathbb{W}_{\bar{i}}$ such that $w \mathcal{E}^i_k w'$, and t_1 holds at w'. As $t_1 \to \varphi$ holds at every world $w' \in \mathbb{W}_{\bar{i}}$, by the semantics of implication, φ holds at w'. That is, for a world w' such that $w \mathcal{E}^i_k w'$, we have that φ holds at w'. Then, by the semantics of $\Diamond_k$, $\Diamond_k \varphi$ holds at w, that is, $\Diamond_k \varphi$ holds. Thus, by the semantics of implication, $t \to \Diamond_a \varphi$ is satisfiable in $\mathbb{W}_i$, that is, $\mathbb{W}_i \models_\ell t \to \Diamond_k \varphi$ and $\mathbb{W}_i \models_g \langle \mathcal{G}_i, \mathcal{K}_i \rangle$. Finally, $\mathbb{C}$ is satisfied.

□

Finally, we show that applying the transformation rules a finite number of times to the problem $\mathbb{C}$ corresponding to a formula φ yields a new problem, $\mathbb{C}'$, that is satisfiable if, and only if, φ is satisfiable. Showing this to the problem $\mathbb{C} = \langle \langle \{\varphi\}, \varnothing \rangle, \langle \varnothing, \varnothing \rangle \rangle$ suffices, as this problem is satisfiable if, and only if, φ is satisfiable.

Theorem 1. *Let φ be an i-formula in NNF, with the corresponding initial $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C}_0 = \langle \langle \mathcal{G}_1, \mathcal{K}_1 \rangle, \langle \mathcal{G}_2, \mathcal{K}_2 \rangle \rangle$ where $\mathcal{G}_i = \{t_0, t_0 \to \varphi\}$ and $\mathcal{K}_i = \mathcal{G}_{\bar{i}} = \mathcal{K}_{\bar{i}} = \varnothing$. Then, the problem $\mathbb{C}'$, obtained after a finite number of applications of the transformation rules given in Section 4.1 is satisfiable if, and only if, $\mathbb{C}$ is satisfiable.*

Proof. By induction on the number of transformation rules applied.

The base case is when φ is a disjunction of literals, then only the rule $\tau_\to$ is applied. The correctness of this step is given by Lemma 7.

The induction hypothesis is that, for every transformation of a problem $\langle \langle \{\varphi\}, \varnothing \rangle, \langle \varnothing, \varnothing \rangle \rangle$ in to a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem in normal form that takes $k < n$ steps, the resulting problem is satisfiable if and only if the first problem is satisfiable.

For the inductive step, consider a chain of transformation rules $(R_1, \ldots, R_n)$ of length $n > 1$, with corresponding problems $(\mathbb{C} = \mathbb{C}_0, \ldots, \mathbb{C}_{n-1}, \mathbb{C}_n = \mathbb{C}')$. We may show that $\mathbb{C}_{n-1}$ is satisfiable if and only if φ is satisfiable by applying the inductive hypothesis to the sub-chain $(\mathbb{C}_0, \dots, \mathbb{C}_{n-1})$.

Now, the last rule, R_n, that transforms $\mathbb{C}_{n-1}$ into $\mathbb{C}_n$, must be one of the rules presented in Section 4.1. We may then conclude this step by applying the corresponding lemma, between Lemmas 5 to 12. With this, we show that $\mathbb{C}_n$ is satisfiable if, and only if, $\mathbb{C}_{n-1}$ is satisfiable. We then have that $\mathbb{C}_n$ is satisfiable if, and only if, φ is satisfiable.

Finally, we have that every transformation of finite length of a problem $\langle\langle\{\varphi\}, \varnothing\rangle, \langle\varnothing, \varnothing\rangle\rangle$ yields a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem that is satisfiable if and only if φ is satisfiable.

$\Box$

5.1.2 Termination of the transformation

In this subsection, we show that the application of the transformation rules terminate and, when it does, we are left with a problem in the normal form.

We start by showing that the transformation rules can be applied if, and only if, the problem is not in the normal form.

Theorem 2. *Given a* $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$*-problem* $\mathbb{C} = \langle\langle\mathcal{G}_1, \mathcal{K}_1\rangle, \langle\mathcal{G}_2, \mathcal{K}_2\rangle\rangle$*, the normal form rules can be applied if, and only if, the problem is not in the normal form.*

Proof. ($\Rightarrow$) We show the contrapositive, that is, if the problem is in the normal form, the rules cannot be applied. We analyse each rule:

(τ_{init}) This rule is only used to construct the initial problem, so it cannot be applied.

($\tau_{\wedge,\vee}$) These rules can only be applied to formulae of the form $t \to \varphi_1 \wedge \varphi_2$ or $t \to \varphi_1 \vee \varphi_2$, and these are not in the clausal form given in Table 4.1.

($\tau_{\to}$) This rule can only be applied to formulae of the form $t \to \varphi_1$, where φ_1 is a disjunction of literals. All the formulae with occurrences of implications in Table 4.1 have a modal operator, so this rule cannot be applied.

(τ_{move}) This rule can only be applied when there are formulae with a modal operator on a $\mathcal{G}_i$-set, which goes against the definition of the normal form. Then, this rule cannot be applied.

($\tau_{\Diamond,\Diamond\!\!\!\!\Diamond}$) These rules can only be applied to formulae of the form $t \to \langle a\rangle\varphi_1$ and $t \to \langle\!\langle a\rangle\!\rangle\varphi_1$, where φ_1 is not a literal, and these are not in the clausal form given in Table 4.1.

($\tau_{\Box,\boxed{\Box}}$) These rules can only be applied to formulae of the form $t \to \boxed{a}\varphi_1$ and $t \to \boxed{\boxed{a}}\varphi_1$, where φ_1 is not a literal, and these are not in the clausal form given in Table 4.1.

Then, we conclude that, as the problem is in normal form, no transformation rule can be applied.

($\Leftarrow$) We must show that, if the problem is not in the normal form, then there is some rule τ that can be applied. We consider only the formulae of the form $t \to \varphi$, as obtained by application of rule τ_{init}. We analyse the possibilities for the formula $t \to \varphi$, on the structure of φ, where φ is in NNF.

(**true**) If the formula is of the form $t \to$ **true**, then we may apply the simplifications in given in Table 4.2 to obtain the formula t, which is in the normal form.

(p) If the formula is of the form $t \to p$, then we may apply the rule $\tau_{\to}$.

($\wedge$) If the formula is of the form $t \to \varphi_1 \wedge \varphi_2$, then we may apply the rule $\tau_{\wedge}$.

($\vee$) If the formula is of the form $t \to \varphi_1 \vee \varphi_2$, then, if $\varphi_1 \vee \varphi_2$ is a disjunction of literals, we may apply the rule $\tau_{\to}$. Otherwise, we may apply rule $\tau_{\vee}$.

($\to$) This is not possible, as the original formulae must be in the NNF, and the rules do not generate nested implications.

($\Diamond, \Diamond\!\!\!\!\Diamond, \Box, \boxed{\Box}$) If the formula has a modality, then the inner formula may not be a literal occurring in the initial problem, otherwise it is in normal form. Then the appropriate $\tau_{\Diamond, \Diamond\!\!\!\!\Diamond, \Box, \boxed{\Box}}$ rule can be applied.

$\Box$

To show the termination property for the transformation rules, we define a *weight* function for problems and show that each transformation rule can only lower the weight of the resulting problem. Intuitively, given a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, the weight function gives information of how much work still needs to be performed in order to obtain a problem in the desired normal form. Before giving the formal definitions, let us consider the following example, where $\mathbb{C}_0$ is an initial problem:

$$\mathbb{C}_0 = \langle \langle \{t_0, t_0 \to p \wedge \boxed{a}^1 \psi\}, \varnothing \rangle, \langle \varnothing, \varnothing \rangle \rangle$$

No transformation rule can be applied to the formula t_0 in $\mathcal{G}_1$, but the transformation rule $\tau_{\wedge}$ can be applied to the implication $t_0 \to p \wedge \boxed{a}^1 \psi$, resulting in the problem $\mathbb{C}_1$:

$$\mathbb{C}_1 = \langle \langle \{t_0, t_0 \to p, t_0 \to \boxed{a}^1 \psi\}, \varnothing \rangle, \langle \varnothing, \varnothing \rangle \rangle$$

At this stage, less transformation steps are needed in order to obtain the normal form of $\mathbb{C}_0$. The initial problem required one step to rewrite the conjunction on the right-hand side of the implication plus all the transformations steps required by the added formulae. The resulting problem will require one less step during the transformation, as the rule $\tau_{\wedge}$ got rid already of the conjunction. The weight function is therefore designed to correspond to the number of steps taken during the transformation procedure. As so, the weight of a problem (set, formula) is basically given by what still needs to be considered for achieving the normal form.

Following the previous example, the weight of $\mathbb{C}_0$ is the weight of transforming $t_0 \to p \wedge \boxed{a}^1 \psi$, which requires to recursively checking how many transformation steps need to be applied to each of the conjuncts. In this case, the weight of the problem $\mathbb{C}_1$ depends on the weights of both introduced formulae. As $\mathbb{C}_1$ is not yet in the normal form, we could, for instance, apply the $\tau_{\Box}$ transformation rule to $t_0 \to \boxed{a}^1 \psi$, resulting in:

$$\mathbb{C}_2 = \langle\langle\{t_0, t_0 \to p, t_1 \to \psi\}, \{t_0 \to \boxed{a}^1 t_1\}\rangle, \langle\varnothing, \varnothing\rangle\rangle$$

Note that $t_0 \to \boxed{a}^1 t_1$ requires no further renaming. Therefore, even if the problem grows in size, the added modal clause does not affect the weight of the problem. In other words, the weight is only affected by further steps required to transform the formula $t_1 \to \psi$ in its normal form. After those examples, we now proceed by providing the definitions and proofs that the transformation into the normal form is indeed terminating and it is correct, that is, it produces a problem in the desired form.

In the following, we will recursively define the weight functions w for problems, sets, and formulae. As no confusion should arise, we will use the same functional symbol for all weight functions. We start by defining the weight of a problem $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ as:

$$w(\mathbb{C}) = w(\mathcal{G}_1) + w(\mathcal{K}_1) + w(\mathcal{G}_2) + w(\mathcal{K}_2)$$

The weight of a set of formulae Γ is given by the sum of the weights of its members:

$$w(\Gamma) = \sum_{\varphi \in \Gamma} w(\varphi)$$

The weight of the union of two sets of formulae, Γ and Γ', is given by the sum of the weights of each set:

$$w(\Gamma \cup \Gamma') = w(\Gamma) + w(\Gamma')$$

The weight function is then defined by case on the structure of formulae. Recall that a formula in a problem can always be simplified to its Negate Normal Form (NNF), by applying the simplification rules given in Table 4.2. Thus, in the following, we only consider those cases. The base cases are those where no transformation rule can be applied (where D is a disjunction of literals, l is a literal, and t, t' are propositional symbols introduced by renaming):

$$w(D) = w(t \to \boxed{a}^i l) = w(t \to \Diamond_a^i l) = w(t \to \boxed{\boxed{\scriptstyle a}}^i t') = w(t \to \Diamond\!\!\!\Diamond_a^i t') = 0$$

If φ is a disjunction of literals, then we also define:

$$w(t \to \varphi) = 1$$

For complex formulae occurring on the right-hand side of implications, the weight function is defined as follows:

$$\begin{array}{rcll}
w(t \to \varphi_1 \wedge \varphi_2) & = & w(t \to \varphi_1) + w(t \to \varphi_2) + 1 & \\
w(t \to \varphi_1 \vee \varphi_2) & = & w(t \to \varphi_1 \vee t_1) + w(t_1 \to \varphi_2) + 1 & \text{where } \varphi_2 \text{ is not a literal} \\
w(t \to \boxed{a}^{i} \varphi) & = & w(t \to \varphi) + 1 & \text{where } \varphi \text{ is not a literal} \\
w(t \to \langle a \rangle^{i} \varphi) & = & w(t \to \varphi) + 1 & \text{where } \varphi \text{ is not a literal} \\
w(t \to \boxed{\boxed{a}}^{i} \varphi) & = & w(t \to \varphi) + 1 & \text{where } \varphi \notin \mathcal{L}it(\Phi) \\
w(t \to \langle\!\langle a \rangle\!\rangle^{i} \varphi) & = & w(t \to \varphi) + 1 & \text{where } \varphi \notin \mathcal{L}it(\Phi)
\end{array}$$

This completes the definition of the weight function.

Recall that we define a transformation into the normal form as a sequence of problems $\mathbb{C}_0$, $\mathbb{C}_1, \ldots$, where $\mathbb{C}_0$ is $start(\varphi)$, $\mathcal{L}it(\Phi)$ is the set of literals occurring in Φ, and each problem $\mathbb{C}_{j+1}$ was obtained from $\mathbb{C}_j$ by an application of the transformation rules given in Section 4.1, for all $j > 0$. In order to show that the transformation rules given in Section 4.1 may only decrease the weight of a problem, we need to show that each possible transformation rule does so, as given by the following lemmas. For a transformation rule τ, we slightly abuse notation and call $w(\tau)$ the difference between the weight of the problem before and after the transformation rule has been applied.

Lemma 13 ($\tau_\wedge$)**.** *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \to \varphi_1 \wedge \varphi_2 \in \mathcal{G}_i$. Let Γ be $\mathcal{G}_i \setminus \{t \to \varphi_1 \wedge \varphi_2\}$ and $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \Gamma \cup \{t \to \varphi_1, t \to \varphi_2\}$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}}$, $\mathcal{K}'_i = \mathcal{K}_i$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying $\tau_\wedge$ to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{array}{rcl}
w(\tau_\wedge) & = & w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
& = & w(\mathcal{G}_i) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
& = & w(\mathcal{G}_i) - w(\mathcal{G}'_i) \\
& = & w(\Gamma) + w(t \to \varphi_1 \wedge \varphi_2) - (w(\Gamma) + w(t \to \varphi_1) + w(t \to \varphi_2)) \\
& = & w(t \to \varphi_1 \wedge \varphi_2) - w(t \to \varphi_1) - w(t \to \varphi_2) \\
& = & w(t \to \varphi_1) + w(t \to \varphi_2) + 1 - w(t \to \varphi_1) - w(t \to \varphi_2) \\
& = & 1
\end{array}$$

□

Lemma 14 ($\tau_\vee$). *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \to \varphi_1 \vee \varphi_2 \in \mathcal{G}_i$. Let Γ be $\mathcal{G}_i \setminus \{t \to \varphi_1 \vee \varphi_2\}$ and $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \Gamma \cup \{t \to \varphi_1 \vee t_1, t_1 \to \varphi_2\}$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}}$, $\mathcal{K}'_i = \mathcal{K}_i$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying $\tau_\vee$ to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{array}{rcl}
w(\tau_\wedge) & = & w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
& = & w(\mathcal{G}_i) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
& = & w(\mathcal{G}_i) - w(\mathcal{G}'_i) \\
& = & w(\Gamma) + w(t \to \varphi_1 \vee \varphi_2) - (w(\Gamma) + w(t \to \varphi_1 \vee t_1) + w(t_1 \to \varphi_2)) \\
& = & w(t \to \varphi_1 \vee \varphi_2) - w(t \to \varphi_1 \vee t_1) - w(t_1 \to \varphi_2) \\
& = & w(t \to \varphi_1 \vee t_1) + w(t_1 \to \varphi_2) + 1 - w(t \to \varphi_1 \vee t_1) - w(t_1 \to \varphi_2) \\
& = & 1
\end{array}$$

□

Lemma 15 ($\tau_\to$). *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \to \varphi \in \mathcal{G}_i$ and φ is a disjunction of literals. Let Γ be $\mathcal{G}_i \setminus \{t \to \varphi\}$ and $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \Gamma \cup \{\neg t \vee \varphi_1\}$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}}$, $\mathcal{K}'_i = \mathcal{K}_i$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying $\tau_\to$ to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{array}{rcl}
w(\tau_\to) & = & w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
& = & w(\mathcal{G}_i) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
& = & w(\mathcal{G}_i) - w(\mathcal{G}'_i) \\
& = & (w(\Gamma) + w(t \to \varphi)) - (w(\Gamma) + w(\neg t \vee \varphi)) \\
& = & w(t \to \varphi) = w(\neg t \vee \varphi) \\
& = & 1 - 0 = 1
\end{array}$$

□

Lemma 16 (τ_{move}). *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \to \varphi \in \mathcal{G}_i$ and φ is a formula whose main operator is a modality. Let $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \mathcal{G}_i$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}}$, $\mathcal{K}'_i = \mathcal{K}_i \cup \{t \to \varphi\}$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying τ_{move} to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{aligned}
w(\tau_{\rightarrow}) &= w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
&= w(\mathcal{G}_i) + w(t \rightarrow \varphi) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - \\
&\quad (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(t \rightarrow \varphi) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
&= 0
\end{aligned}$$

□

Lemma 17 ($\tau_{\Box}$)**.** *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \rightarrow \boxed{a}^i \varphi \in \mathcal{K}_i$ and φ is not a literal. Let Γ be $\mathcal{K}_i \setminus \{t \rightarrow \boxed{a}^i \varphi\}$ and $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \mathcal{G}_i \cup \{t_1 \rightarrow \varphi\}$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}}$, $\mathcal{K}'_i = \Gamma \cup \{t \rightarrow \boxed{a}^i t_1\}$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying $\tau_{\Box}$ to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{aligned}
w(\tau_{\Box}) &= w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
&= w(\mathcal{G}_i) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
&= w(\mathcal{G}_i) + w(\mathcal{K}_i) - (w(\mathcal{G}'_i) - w(\mathcal{K}'_i)) \\
&= w(\mathcal{G}_i) + w(\Gamma) + w(t \rightarrow \boxed{a}^i \varphi) - \\
&\quad (w(\mathcal{G}_i) + w(t_1 \rightarrow \varphi) + w(\Gamma) + w(t \rightarrow \boxed{a}^i t_1)) \\
&= w(t \rightarrow \boxed{a}^i \varphi) - (w(t_1 \rightarrow \varphi) + w(t \rightarrow \boxed{a}^i t_1)) \\
&= w(t \rightarrow \boxed{a}^i \varphi) - w(t_1 \rightarrow \varphi) \\
&= w(t \rightarrow \varphi) + 1 - w(t_1 \rightarrow \varphi) \\
&= 1
\end{aligned}$$

□

Lemma 18 ($\tau_{\Diamond}$)**.** *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \rightarrow \langle a \rangle^i \varphi \in \mathcal{K}_i$ and φ is not a literal. Let Γ be $\mathcal{K}_i \setminus \{t \rightarrow \langle a \rangle^i \varphi\}$ and $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \mathcal{G}_i \cup \{t_1 \rightarrow \varphi\}$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}}$, $\mathcal{K}'_i = \Gamma \cup \{t \rightarrow \langle a \rangle^i t_1\}$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying $\tau_{\Diamond}$ to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{aligned}
w(\tau_{\Diamond}) &= w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
&= w(\mathcal{G}_i) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
&= w(\mathcal{G}_i) + w(\mathcal{K}_i) - (w(\mathcal{G}'_i) - w(\mathcal{K}'_i)) \\
&= w(\mathcal{G}_i) + w(\Gamma) + w(t \rightarrow \langle a \rangle^i \varphi) - (w(\mathcal{G}_i) + w(t_1 \rightarrow \varphi) + w(\Gamma) + w(t \rightarrow \langle a \rangle^i t_1)) \\
&= w(t \rightarrow \langle a \rangle^i \varphi) - (w(t_1 \rightarrow \varphi) + w(t \rightarrow \langle a \rangle^i t_1)) \\
&= w(t \rightarrow \langle a \rangle^i \varphi) - w(t_1 \rightarrow \varphi) \\
&= w(t \rightarrow \varphi) + 1 - w(t_1 \rightarrow \varphi) \\
&= 1
\end{aligned}$$

$\square$

Lemma 19 ($\tau_{\boxed{}}$). *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \to \boxed{a}^i \varphi \in \mathcal{K}_i$ and φ is not in $\mathcal{L}it(\Phi)$. Let Γ be $\mathcal{K}_i \setminus \{t \to \boxed{a}^i \varphi\}$ and $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \mathcal{G}_i$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}} \cup \{t_1 \to \varphi\}$, $\mathcal{K}'_i = \Gamma \cup \{t \to \boxed{a}^i t_1\}$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying $\tau_{\boxed{}}$ to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{array}{rcl}
w(\tau_{\boxed{}}) & = & w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
& = & w(\mathcal{G}_i) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
& = & w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_i) - (w(\mathcal{G}'_{\bar{i}}) - w(\mathcal{K}'_i)) \\
& = & w(\mathcal{G}_{\bar{i}}) + w(\Gamma) + w(t \to \boxed{a}^i \varphi) - (w(\mathcal{G}_{\bar{i}}) + w(t_1 \to \varphi) + w(\Gamma) + w(t \to \boxed{a}^i t_1)) \\
& = & w(t \to \boxed{a}^i \varphi) - (w(t_1 \to \varphi) + w(t \to \boxed{a}^i t_1)) \\
& = & w(t \to \boxed{a}^i \varphi) - w(t_1 \to \varphi) \\
& = & w(t \to \varphi) + 1 - w(t_1 \to \varphi) \\
& = & 1
\end{array}$$

$\square$

Lemma 20 ($\tau_{\Diamond}$). *Let $\mathbb{C}_j = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \to \Diamond_a^i \varphi \in \mathcal{K}_i$ and φ is not in $\mathcal{L}it(\Phi)$. Let Γ be $\mathcal{K}_i \setminus \{t \to \Diamond_a^i \varphi\}$ and $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ be a problem such that $\mathcal{G}'_i = \mathcal{G}_i$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}} \cup \{t_1 \to \varphi\}$, $\mathcal{K}'_i = \Gamma \cup \{t \to \Diamond_a^i t_1\}$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. That is, $\mathbb{C}_{j+1}$ is the result of applying $\tau_{\Diamond}$ to $\mathbb{C}_j$. Then, $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$.*

Proof.

$$\begin{array}{rcl}
w(\tau_{\Diamond}) & = & w(\mathbb{C}_j) - w(\mathbb{C}_{j+1}) \\
& = & w(\mathcal{G}_i) + w(\mathcal{K}_i) + w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_{\bar{i}}) - (w(\mathcal{G}'_i) + w(\mathcal{K}'_i) + w(\mathcal{G}'_{\bar{i}}) + w(\mathcal{K}'_{\bar{i}})) \\
& = & w(\mathcal{G}_{\bar{i}}) + w(\mathcal{K}_i) - (w(\mathcal{G}'_{\bar{i}}) - w(\mathcal{K}'_i)) \\
& = & w(\mathcal{G}_{\bar{i}}) + w(\Gamma) + w(t \to \Diamond_a^i \varphi) - (w(\mathcal{G}_{\bar{i}}) + w(t_1 \to \varphi) + w(\Gamma) + w(t \to \Diamond_a^i t_1)) \\
& = & w(t \to \Diamond_a^i \varphi) - (w(t_1 \to \varphi) + w(t \to \Diamond_a^i t_1)) \\
& = & w(t \to \Diamond_a^i \varphi) - w(t_1 \to \varphi) \\
& = & w(t \to \varphi) + 1 - w(t_1 \to \varphi) \\
& = & 1
\end{array}$$

$\square$

Lemmas 13-20 show that each step of the transformation lowers the weight of a given problem until the desired normal form is achieved. By induction on the number of steps of a transformation, we get the following result.

Theorem 3 (Termination of the transformation)**.** *Let $\tau = (\mathbb{C}_0, \mathbb{C}_1, \ldots)$ be a transformation. Then, there is n such that $\mathbb{C}_n \in \tau$ and it is in the normal form.*

Proof. By definition, τ is a sequence of problems where $\mathbb{C}_0$ is $start(\varphi)$, $\mathcal{L}it(\Phi)$ is the set of literals not occurring in φ, and each problem $\mathbb{C}_{j+1}$ was obtained from $\mathbb{C}_j$ by an application of the transformation rules given in Section 4.1, for all $j > 0$. By Lemmas 13-20, for every $j > 0$, we have that $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$. We now show that we can only apply the transformation rules if the weight of a problem is positive.

We use the contrapositive and show that if the weight is not positive, then no transformation rule can be applied. As defined, the weight function only assigns values greater or equal to zero. If a problem has weight zero, we can no further apply any transformation rule, as by definition this means that all clauses in all sets of a problem are either a literal clause, an a_i-clause, or an $\mathcal{E}_i^k$-clause. Note that in almost all cases, the inequality $w(\mathbb{C}_j) \geq w(\mathbb{C}_{j+1})$ is actually strict, that is, we have that $w(\mathbb{C}_j) > w(\mathbb{C}_{j+1})$. The only exception is when τ_{move} is applied. In this case, assume that $\mathbb{C}_j$ is of the form $\langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem, where $t \rightarrow \varphi \in \mathcal{G}_i$ and φ is a formula whose main operator is a modality. Let Γ be $\mathcal{G}_i \setminus \{t \rightarrow \varphi\}$. Then, after the application of τ_{move}, we obtain $\mathbb{C}_{j+1} = \langle\langle \mathcal{G}'_1, \mathcal{K}'_1\rangle, \langle \mathcal{G}'_2, \mathcal{K}'_2\rangle\rangle$ where $\mathcal{G}'_i = \Gamma$, $\mathcal{G}'_{\bar{i}} = \mathcal{G}_{\bar{i}}$, $\mathcal{K}'_i = \mathcal{K}_i \cup \{t \rightarrow \varphi\}$, and $\mathcal{K}'_{\bar{i}} = \mathcal{K}_{\bar{i}}$. We assume, without loss of generality, there is only one formula of the form $t \rightarrow \varphi \in \mathcal{G}_i$ and φ is a formula whose main operator is a modality. There are now two cases to consider: either $t \rightarrow \varphi$ is already in the normal form or one of the transformation rules for modalities can be applied. If $t \rightarrow \varphi$ is already in the normal form and $w(\mathbb{C}_{j+1}) = 0$, then $n = j + 1$ and we are done. If $t \rightarrow \varphi$ is already in the normal form and $w(\mathbb{C}_{j+1}) > 0$, then, by Lemmas 13-15, there is another transformation rule that can be applied whose resulting weight $w(\mathbb{C}_{j+2})$ is strictly lower than $w(\mathbb{C}_{j+1})$. Hence, $w(\mathbb{C}_j) > w(\mathbb{C}_{j+2})$. If $t \rightarrow \varphi$ is not in the normal form, by Lemmas 17-20, an application of any of the transformation rules $\tau_{\Box}$, $\tau_{\Diamond}$, $\tau_{\boxed{\Box}}$ or $\tau_{\diamond\!\!\!\Diamond}$ to $t \rightarrow \varphi$ in $\mathcal{K}'_i$ will result in a problem $\mathbb{C}_{j+2}$ such that $w(\mathbb{C}_{j+1}) > w(\mathbb{C}_{j+2})$. Again, it follows that $w(\mathbb{C}_j) > w(\mathbb{C}_{j+2})$. This suffices to show that the weight of problems will eventually decrease as desired. As a problem is finite, the number of times we can decrease the weight of problems in a transformation is also finite. Therefore, there is n such that $\mathbb{C}_n \in \tau$, $w(\mathbb{C}_n) = 0$, that is, $\mathbb{C}_n$ is in the normal form, and the transformation procedure terminates. $\square$

5.2 Calculus

This section contains the proofs of soundness, termination and completeness, together with the necessary lemmas.

5.2.1 Soundness of the calculus

The proof of soundness is done in a straightforward way. We prove soundness of rules [$\mathcal{E}$-**GEN1**] and [$\mathcal{E}$-**GEN2**], and then show soundness of the calculus itself by induction on the length of a derivation.

Lemma 21. $\mathcal{E}$-**GEN1** *is sound. That is, for a problem* $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$*, where* $\mathbb{P} = \{l'_1 \to \boxed{k}^{\,i}\neg l_1, l'_m \to \boxed{k}^{\,i}\neg l_m, l' \to \langle k\rangle^{i}\neg l\} \subseteq \mathcal{K}_i$ *and* $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ *is satisfiable, if* $\mathcal{G}_{\bar{i}} \cup \mathcal{K}_{\bar{i}} \vdash_{\mathsf{C}_{\bar{i}}} l_1 \vee \cdots \vee l_m \vee l$*, then we have that* $\mathcal{C}_i \models \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'$.

Proof. As $\mathcal{C}_i$ is satisfiable, there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_i \models_g \mathbb{P}$. Let w be a world in $\mathbb{W}_i$. For the purpose of contradiction, suppose $(\mathbb{W}_i, w) \models_\ell l'_1 \wedge \ldots \wedge l'_m \wedge l'$.

As $\mathbb{P}$ is globally satisfiable, then we have that $(\mathbb{W}_i, w) \models_\ell \{l'_1 \to \boxed{k}^{\,i}\neg l_1, l'_m \to \boxed{k}^{\,i}\neg l_m, l' \to \langle k\rangle^{i}\neg l\}$. By the semantics of conjunction and the semantics of implication, we have that $(\mathbb{W}_i, w) \models_\ell \{\boxed{k}^{\,i}\neg l_1,\ \ldots,\ \boxed{k}^{\,i}\neg l_m,\ \langle k\rangle^{i}\neg l\}$. By the semantics of $\langle k\rangle^{i}$, there is a world $w' \in \mathbb{W}_{\bar{i}}$ such that $w\mathcal{E}^i_k w'$ and $(\mathbb{W}_{\bar{i}}, w') \models_\ell \neg l$. By the semantics of $\boxed{k}^{\,i}$, as $w\mathcal{E}^i_k w'$, we also have that $(\mathbb{W}_{\bar{i}}, w') \models_\ell \{\neg l_1,\ \ldots,\ \neg l_m\}$.

From $\mathcal{G}_{\bar{i}} \cup \mathcal{K}_{\bar{i}} \vdash_{\mathsf{C}_{\bar{i}}} l_1 \vee \cdots \vee l_m \vee l$, as $\mathsf{C}_{\bar{i}}$ is strongly correct, we have that $\mathcal{G}_{\bar{i}} \cup \mathcal{K}_{\bar{i}} \models_g l_1 \vee \cdots \vee l_m \vee l$. This means that we have $(\mathbb{W}_{\bar{i}}, w') \models_\ell l_1 \vee \cdots \vee l_m \vee l$, which contradicts with $(\mathbb{W}_{\bar{i}}, w') \models_\ell \{\neg l_1,\ \ldots,\ \neg l_m,\ \neg l\}$, as a literal and its negation cannot be satisfied at the same time. Thus, our assumption that $(\mathbb{W}_i, w) \models_\ell l'_1 \wedge \ldots \wedge l'_m \wedge l'$ is incorrect, that is, there cannot be a world where all these literals hold. This means that at least one $\neg l'_i$ must hold in $\mathbb{W}_i$, that is, $\mathbb{W}_i \models_g \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'$, and the conclusion of the rule is satisfied. Then, $\mathcal{E}$-**GEN1** is sound. □

Lemma 22. $\mathcal{E}$-**GEN2** *is sound. That is, for a problem* $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$*, where* $\mathbb{P} = \{l'_1 \to \boxed{k}^{\,i}\neg l_1, l'_m \to \boxed{k}^{\,i}\neg l_m, l' \to \langle k\rangle^{i}\neg l\} \subseteq \mathcal{K}_i$ *and* $\mathcal{C}_i = \langle \mathcal{G}_i, \mathcal{K}_i\rangle$ *is satisfiable, if* $\mathcal{G}_{\bar{i}} \cup \mathcal{K}_{\bar{i}} \vdash_{\mathsf{C}_{\bar{i}}} l_1 \vee \cdots \vee l_m$*, then we have that* $\mathcal{C}_i \models \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'$.

Proof. As $\mathcal{C}_i$ is satisfiable, there is a model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}^1_j)_{j\in\mathcal{I}_1}, (\mathcal{E}^2_k)_{k\in\mathcal{I}_2}\rangle$, with $\mathbb{W}_i \models_g \mathbb{P}$. Let w be a world in $\mathbb{W}_i$. For the purpose of contradiction, suppose $(\mathbb{W}_i, w) \models_\ell l'_1 \wedge \ldots \wedge l'_m \wedge l'$.

As $\mathbb{P}$ is globally satisfiable, then we have that $(\mathbb{W}_i, w) \models_\ell \{l'_1 \to \boxed{k}^{\,i}\neg l_1, l'_m \to \boxed{k}^{\,i}\neg l_m, l' \to \langle k\rangle^{i}\neg l\}$. By the semantics of conjunction and the semantics of implication, we have that $(\mathbb{W}_i, w) \models_\ell \{\boxed{k}^{\,i}\neg l_1,\ \ldots,\ \boxed{k}^{\,i}\neg l_m,\ \langle k\rangle^{i}\neg l\}$. By the semantics of $\langle k\rangle^{i}$, there is a world $w' \in \mathbb{W}_{\bar{i}}$ such that $w\mathcal{E}^i_k w'$ and $(\mathbb{W}_{\bar{i}}, w') \models_\ell \neg l$. By the semantics of $\boxed{k}^{\,i}$, as $w\mathcal{E}^i_k w'$, we also have that $(\mathbb{W}_{\bar{i}}, w') \models_\ell \{\neg l_1,\ \ldots,\ \neg l_m\}$.

As $\mathsf{C}_{\bar{i}}$ is strongly correct, we have that $\mathcal{G}_{\bar{i}} \cup \mathcal{K}_{\bar{i}} \models_g l_1 \vee \cdots \vee l_m$. This means that we have $(\mathbb{W}_{\bar{i}}, w') \models_\ell l_1 \vee \cdots \vee l_m$, which contradicts with $(\mathbb{W}_{\bar{i}}, w') \models_\ell \{\neg l_1,\ \ldots,\ \neg l_m\}$, as a literal and its negation cannot be satisfied at the same time. Thus, our assumption that $(\mathbb{W}_i, w) \models_\ell l'_1 \wedge \ldots \wedge l'_m \wedge l'$ is incorrect, that is, there cannot be a world where all these literals hold. This means that at least one $\neg l'_i$ must hold in $\mathbb{W}_i$, that is, $\mathbb{W}_i \models_g \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l$, and the conclusion of the rule is satisfied. Then, $\mathcal{E}$-**GEN2** is sound. $\square$

Theorem 4 (Correctness of $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$). *Let* $(\mathbb{C}_1, \mathbb{C}_2, \ldots)$ *be a derivation. Then, for all* $n \in \mathbb{N}$, $\mathbb{C}_n$ *is satisfiable if* $\mathbb{C}_1$ *is satisfiable. That is,* $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ *is sound.*

Proof. By induction on n. The base case is when no rule is applied, that is, the derivation has only one problem, $\mathbb{C}_1$. This case is trivial, as $\mathbb{C}_1$ is satisfiable if itself is satisfiable.

For the inductive step, we have as hypothesis that $\mathbb{C}_{n+1}$ is satisfiable if $\mathbb{C}_n$ is satisfiable, and that problem $\mathbb{C}_i$ itself is satisfiable. We need to show that $\mathbb{C}_{n+1}$ is satisfiable.

$\mathbb{C}_{i+n}$ is obtained from $\mathbb{C}_n$ by an application of either $\mathcal{E}$-**GEN1** or $\mathcal{E}$-**GEN2**. That is, $\mathbb{C}_{n+1} = \langle\langle \mathcal{G}_1^n \cup \{D\}, \mathcal{K}_1^n\rangle, \langle \mathcal{G}_2^n, \mathcal{K}_2^n\rangle\rangle$ or $\mathbb{C}_{n+1} = \langle\langle \mathcal{G}_1^n, \mathcal{K}_1^n\rangle, \langle \mathcal{G}_2^n \cup \{D\}, \mathcal{K}_2^n\rangle\rangle$, where D is the result of the application of the rule. Let $\mathcal{C}_n = \langle \mathcal{G}_i^n, \mathcal{K}_i^n\rangle$ be the component of $\mathbb{C}_n$ to which the rule is applied. Then, $\mathcal{C}_{n+1} = \langle \mathcal{G}_1^n \cup \{D\}, \mathcal{K}_i^n\rangle$ is the resulting component. By Lemmas 21 and 22, $\mathcal{C}_n \models_g D$. This means that $\mathbb{C}_{n+1}$ is also satisfiable.

This concludes the inductive proof of the theorem, and we have that $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ is sound. $\square$

The following proof shows that the number of clauses that can be generated by the resolution procedure is finite. This means that the $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem will eventually get *saturated*, that is, no rule can be applied which generates new clauses. This shows that the length of a derivation is always finite.

Theorem 5 (Termination of $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$). *Let* $\mathbb{C}$ *be a* $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$*-problem and* $(\mathbb{C}_0, \mathbb{C}_1, \ldots)$ *be a derivation by* $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$*. Then, there exists a natural number* k *such that* $\mathbb{C}_k$ *is saturated or contains a contradiction. That is, the resolution calculus* $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ *is terminating.*

Proof. Let n denote the number of literals in a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C}_0$. Note that the inference rules do not add new literals in their conclusion. This means that no new literals are created in the derivation. Rules $\mathcal{E}$-**GEN1** and $\mathcal{E}$-**GEN2** always add clauses, but the number of such clauses is limited by the number of literals from which they can be created, as these are always propositional clauses. We identify a clause with its set of literals, and work applying simplification whenever possible. This way, no clause may contain both its literal and its negation, while also allowing us to associate a clause with its set of literals. This also allow us to represent the set of all possible clauses by $2^{\mathcal{L}it(\mathbb{C}_0)}$, where $\mathcal{L}it(\mathbb{C}_0)$ is the set of literals in the initial problem. Then, the maximal number of possible clauses is equal to $|2^{\mathcal{L}it(\mathbb{C}_0)}|$. As $|\mathcal{L}it(\mathbb{C}_0)| = n$, then $|2^{\mathcal{L}it(\mathbb{C}_0)}| = 2^n$. Thus, we have that the number of clauses that can be derived by $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ is at most 2^n. Finally, we have that either a derivation ends in a contradiction or, eventually, no new clauses can be derived, which means that $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ always terminates. $\square$

5.2.2 Completeness of the calculus

In this subsection, we prove that the calculus $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ is complete. This is done by first defining a structure representing all possible models, called a *connected graph*. Each node of this graph represent a consistent set of literals, and we give a definition for what it means for a node to satisfy a formula. We prove completeness by showing that if a formula is unsatisfiable, the formula does not hold in any node of the graph, causing us to remove each node until the graph becomes empty. Then, we show that application of the inference rules of $\mathsf{RES}_{\mathcal{E}(\mathsf{C1},\,\mathsf{C2})}$ can mimic this removal of nodes.

First, we begin with a definition of satisfiability of a formula in a set of literals and, based on this definition, satisfiability on a node.

Definition 28. Let $\mathcal{V}$ be a consistent set of literals and modal literals. Let φ, φ' and ψ be boolean combinations of literals and modal literals. We say that $\mathcal{V}$ satisfies φ, (written as $\mathcal{V} \models \varphi$) if, and only if:

- $\varphi \in \mathcal{V}$, for a literal or modal literal φ;
- φ is of the form $\varphi \wedge \psi$, $\mathcal{V} \models \varphi$ and $\mathcal{V} \models \psi$
- φ is of the form $\varphi \vee \psi$, $\mathcal{V} \models \varphi$ or $\mathcal{V} \models \psi$
- φ is of the form $\neg\psi$ and $\mathcal{V}$ does not satisfy ψ (written as $\mathcal{V} \not\models \psi$)

Definition 29. Let $\mathcal{V}$ be a consistent set of literals and modal literals, η be a node that contains all the literals and modal literals in $\mathcal{V}$, φ be a boolean combination of literals and modal literals, and $\mathcal{X} = \{\varphi_1, \ldots, \varphi_m\}$ be a set of formulae, where each φ_i, $i \leq i \leq m$, is a boolean combination of literals and modal literals. We say that η satisfies φ (written as $\eta \models \varphi$) if, and only if, $\mathcal{V} \models \varphi$. We say that η satisfies $\mathcal{X}$ (written as $\eta \models \mathcal{X}$) if, and only if, $\eta \models \varphi_1 \wedge \ldots \wedge \varphi_m$.

We may now define the graph structures.

Definition 30. A component graph $\mathbb{G}_i = \langle \mathcal{N}_i, \mathcal{R}_i^1, \ldots, \mathcal{R}_i^n \rangle$, where $\mathcal{N}_i$ is a set of nodes and each $\mathcal{R}_i^j$ is a set of edges labelled by $a \in \mathcal{A}_i$, is built from the set of i-clauses in a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C}$. Each node in $\mathcal{N}$ corresponds to a set of i-literals and i-modal literals in $\mathbb{C}$, and each set of edges corresponds to the accessibility relations of the agents in $\mathcal{A}_i$.

Definition 31. A connected graph $\mathbb{G} = \langle \mathbb{G}_1, \mathbb{G}_2, E_1^1, \ldots, E_i^n, E_2^1, \ldots, E_2^m \rangle$ is built from the component graphs $\mathbb{G}_1$ and $\mathbb{G}_2$ of a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem $\mathbb{C}$. The sets E_i^j correspond to the connecting accessibility relations of $\mathcal{E}$.

We now define how these graphs are constructed. We assume that the calculi for the component logics L_1 and L_2 are strongly complete, and assume that we can use their own graph-building procedures that generate the component graphs $\mathbb{G}_i$, with the nodes in $\mathcal{N}_i$ being maximal consistent sets of i-literals and i-modal literals, and the edges in $\mathcal{R}_i^j$ connecting theses nodes. These graphs are consistent in the sense that all the nodes must satisfy the i-clauses in $\mathbb{C} = \langle\langle \mathcal{G}_1, \mathcal{K}_1\rangle, \langle \mathcal{G}_2, \mathcal{K}_2\rangle\rangle$, as they were given by the procedures of the strongly complete calculi for the logics L_1 and L_2.

With the component graph $\mathbb{G}_i$ already given, we construct the connected graph $\mathbb{G}$ by first connecting both graphs with the connecting edges E_i^j, as follows. For each pair $\eta \in \mathbb{G}_i$ and $\eta' \in \mathbb{G}_{\bar{i}}$, there is a j-edge from η to η'. Then, for every node η, if $(l' \to \boxed{j}^{i} l) \in \mathcal{K}_i$ and $\eta \models l'$, delete any j-edges from η to η' such that $\eta' \not\models l$. This ensures that all the positive j-clauses are satisfied by any nodes in $\mathbb{G}$. Next, consider any nodes that do not satisfy the negative j-clauses in $\mathcal{K}_i$. For each node η and each $j \in \mathcal{E}_i$, if $(l' \to \neg\boxed{j}^{i} l) \in \mathcal{K}_i$, $\eta \models l'$ and there is no j-edge between η and a node that satisfies $\neg l$, then η is deleted. This ensures that all negative j-clauses are satisfied by all nodes in $\mathbb{G}$. Note that, after this step, we must use the completeness proofs from the component logics to, again, delete any nodes in which the negative i-modal clauses are no longer satisfied, as we deleted some nodes.

The connected graph obtained after performing all possible deletions is called *reduced behaviour graph*. We say that this connected graph is non-empty if $\mathbb{G}_1$ is non-empty.

We first show that a set of clauses is satisfiable if, and only if, the reduced graph for the corresponding $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem in normal form is non-empty.

Lemma 23. *Let $\mathbb{C}$ be a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem. $\mathbb{C}$ is satisfiable in $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$ if, and only if, the reduced behaviour graph $\mathbb{G}$ constructed from $\mathbb{C}$ is non-empty.*

Proof. ($\Rightarrow$) Assume that $\mathbb{C}$ is satisfiable. If we construct a graph $\mathbb{G}$ from $\mathbb{C}$, the component graphs $\mathbb{G}_i$ are non-empty, as the calculus for each logic L_i is strongly correct. After the construction of the component graphs, nodes are deleted only if negative j-clauses cannot be satisfied. As $\mathbb{C}$ is satisfiable, at least the 1-formulae can be satisfied. Hence, a satisfiable problem will result in a non-empty graph.

($\Leftarrow$) Assume that the reduced graph $\mathbb{G} = \langle \mathbb{G}_1, \mathbb{G}_2, E_1^1, \dots, E_i^n, E_2^1, \dots, E_2^m\rangle$ constructed from $\mathbb{C}$ is non-empty. First consider the case where $\mathbb{G}_2$ is also non-empty. To show that $\mathbb{C}$ is satisfiable, we construct a model $\mathbb{M}$ from $\mathbb{G}$. By the completeness proofs of the calculi for L_1 and L_2, we may construct two models $\mathbb{W}_1 = \langle \mathcal{W}^1, \pi_1, \mathcal{R}_1^1, \dots, \mathcal{R}_{n_1}^1\rangle$ and $\mathbb{W}_2 = \langle \mathcal{W}^2, \pi_2, \mathcal{R}_1^2, \dots, \mathcal{R}_{n_2}^2\rangle$ from $\mathbb{G}_1$ and $\mathbb{G}_2$ in such a way that there are two functions $node_1 : \mathcal{N}_1 \to \mathcal{W}_1$ and $node_2 : \mathcal{N}_2 \to \mathcal{W}_2$ mapping each consistent set of literals and modal literals to worlds, where each node is assigned to a different world, and if two nodes are connected through and a-edge, then the corresponding worlds are connected through the $\mathcal{R}_i^a$. Now, all we have to do to finish the construction of the model $\mathbb{M} = \langle \mathbb{W}_1, \mathbb{W}_2, (\mathcal{E}_j^1)_{j \in \mathcal{I}_1}, (\mathcal{E}_k^2)_{k \in \mathcal{I}_2}\rangle$ is to construct each relation $\mathcal{E}_j^i$ connecting the worlds that are connected by the respective E_j^i-edges.

For the case where $\mathbb{G}_2$ is empty, as $\mathbb{G}_1$ has at least one node, this means that the second component is unreachable. Then, we may construct the model similarly to the previous case, with the difference that the model $\mathbb{W}_2$ may be any model and the $\mathcal{E}^1$-relations are empty. Then, the second component is unreachable and the satisfiability of the problem depends only on the model $\mathbb{M}_1$. Then, $\mathbb{M} \models_g \mathbb{C}$. $\square$

Theorem 6. *Let $\mathbb{C}$ be an unsatisfiable $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem. A refutation can be derived by applying the resolution rules for* $\mathsf{RES}_{\mathcal{E}(\mathsf{C}_1,\,\mathsf{C}_2)}$.

Proof. We proceed by case analysis. First, assume that the literal 1-clauses are unsatisfiable, which means that all nodes on the first component will be removed. Then, by completeness of C_1, $\mathcal{G}_1 \cup \mathcal{K}_1 \vdash_{\mathsf{C}_1}$ **false**. By the definition of derivation, $\mathsf{RES}_{\mathcal{E}(\mathsf{C}_1,\,\mathsf{C}_2)}$ derives **false**. The case where the set of literal 2-clauses are unsatisfiable, that is, when all the nodes of the second component are removed, would only imply unsatisfiability of the whole problem if a negative $\mathcal{E}_1^j$-clause $(l \to \langle j\rangle^{1} l')$ has a satisfiable left-hand side in some node $\eta \in \mathbb{G}_1$. If this is the case, as there are no nodes in the second component, the negative $\mathcal{E}_1^j$-clause cannot be satisfied, and the node η is removed. We can mimic this steps by first observing that the completeness of the calculus for the second component, together with its unsatisfiability, means that its calculus can derive $\neg l'$ in any node. This means that $\mathcal{G}_2 \cup \mathcal{K}_2 \vdash_{\mathsf{C}_2} \neg l'$. Then, we may apply $\mathcal{E}$**-GEN1** to derive $\neg l$. Then, by the completeness of C_1, **false** is derived in the first component and, by the definition of a derivation, $\mathsf{RES}_{\mathcal{E}(\mathsf{C}_1,\,\mathsf{C}_2)}$ derives **false**. From now on, we may assume that the sets of literal clauses are by themselves satisfiable.

Next, if the non-reduced graph is not empty, consider any nodes that do not satisfy the negative $\mathcal{E}_i^j$-clauses in $\mathbb{C}$. For each node η and for each relation E_i^j, if $(l \to \langle j\rangle l')$ is in $\mathbb{C}$, $\eta \models l$ and there is no $\mathcal{E}_i^j$-edge between η and a node that satisfies l', then η is deleted.

Let $\mathbb{C}_j^\eta$ be the set of positive $\mathcal{E}_i^j$-clauses in $\mathbb{C}$, that is, clauses of the form $(l_j \to \boxed{j}^{i} l'_j)$, whose left-hand side are satisfiable by η. Let $\mathbb{R}_j^\eta$ be the set of literals in the scope of $\boxed{j}$ in the clauses of $\mathbb{C}_j^\eta$, that is, if $(l_j \to \boxed{j}^{i} l'_j)$, then $l'_j \in \mathbb{R}_j^\eta$. From the construction of the graph, for a clause $(l \to \langle j\rangle^{i} l')$, if $\eta \models l$ but there is no $\mathcal{E}_i^j$-edge to a node containing l', it means that $l', \mathbb{R}_j^\eta$ and the literal clauses of the $\bar{i}$-component must be contradictory. As l' alone is not contradictory and we are assuming that the literal clauses are satisfiable, we have the following cases:

1. Assume that $\mathbb{R}_j^\eta$ itself is contradictory. This means there must be clauses of the form $(l_1 \to \boxed{j}^{i} l'')$, $(l_2 \to \boxed{j}^{i} \neg l'') \in \mathbb{C}_j^\eta$, where $\eta \models l_1$ and $\eta \models l_2$. Thus, we can apply $\mathcal{E}$**-GEN2** to the clauses and the negative modal clause $(l \to \langle j\rangle^{i} l')$, deriving $(\neg l_1 \lor \neg l_2)$. This rule may be applied, as $(l'' \lor \neg l'')$ is a tautology and C_i is strongly complete. The addition of this resolvent means η will be deleted as required.

2. Assume that l' and $\mathbb{R}_j^\eta$ is contradictory. Then, $\mathbb{C}_j^\eta$ contains a clause $(l_1 \to \boxed{j}^{i} \neg l')$ where, from the definition of $\mathbb{C}_j^\eta$, $\eta \models l_1$. Thus, by an application of $\mathcal{E}$**-GEN1** to this clause and $(l \to \langle j\rangle^{i} l')$, we derive $(\neg l_1 \lor \neg l)$. This rule may be applied, as $(l' \lor \neg l')$ is a tautology and C_i is strongly complete. The addition of this resolvent means η will be deleted as required.

3. Assume that l' and the literal clauses of the $\bar{\imath}$-component are contradictory. As the calculus for $\mathsf{C}_{\bar{\imath}}$ is strongly complete, it derives $\neg l'$. This means we may apply $\mathcal{E}$**-GEN1** to generate $\neg l$ as a resolvent, which will delete η as required.

4. Assume that $\mathbb{R}^{\eta}_{j}$ and the literal clauses of the $\bar{\imath}$-component are contradictory, but not l'. Let $l_1, \ldots, l_n$ be the relevant literals in $\mathbb{R}^{\eta}_{j}$. As $l_1 \wedge \ldots \wedge l_n$ and the literal clauses of the $\bar{\imath}$-component are contradictory, and as $\mathsf{C}_{\bar{\imath}}$ is strongly complete, we have that $\mathsf{C}_{\bar{\imath}}$ derives $\neg l_1 \vee \ldots \vee \neg l_n$, and then we may apply $\mathcal{E}$**-GEN2** to the relevant clauses which will delete η as required.

5. Assume that l', $\mathbb{R}^{\eta}_{j}$ and the literal clauses all contribute to the contradiction. We may proceed in a way similar to the above. As $\mathsf{C}_{\bar{\imath}}$ is strongly complete, it may derive $\neg l_1 \vee \ldots \vee \neg l_n \vee \neg l'$, and then we may apply $\mathcal{E}$**-GEN2** to delete η, as required.

Essentially, we may mimic the deletion of nodes by applying rules $\mathcal{E}$**-GEN1** and $\mathcal{E}$**-GEN2** to transfer knowledge between the components and generate the resolvents, which are simple enough to generate a contradiction directly in the component. We leave the cases where an inconsistency is caused by i-literal clauses to C_{i}, and focus on the cases where knowledge from both components is needed. We use $\mathcal{E}$**-GEN1** when the contradiction involves the literal inside the relevant negative clause, and $\mathcal{E}$**-GEN2** when it does not.

$\square$

Chapter 6

Conclusion and future work

This work presented a modalised version of $\mathcal{E}$-connections, following the approach in [16]. We have also proposed a normal form for a set of $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-formulae, which separates the different domains of knowledge as much as possible, facilitating the development of inference rules that are specialised to deal with the connections only, without doing domain-specific reasoning. We showed that the transformation process is both correct and terminating, in the sense that it always produces a correspondent $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem in normal form, that is satisfiable if, and only if, the original set of formulae is satisfiable.

Our definition of a $\mathcal{C}^{\mathcal{M}}(\mathsf{L}_1, \mathsf{L}_2)$-problem allowed us to present a modular calculus for $\mathcal{E}$-connected logics, together with proofs for correctness, completeness and termination. We have also made available an algorithm and a prototype implementation for this calculus, which rely on external provers for the component logics to deal with domain-specific knowledge.

Some of the aspects of the problem that a production-ready prover might have to overcome are discussed. We presented specific parts of our implementation that can be improved, from the optimization of the query-clause generation to the possibility of defining a basic protocol for communication, not only allowing better modularity of the components, but also specifying requirements that their provers have to meet to allow for efficient processing of the queries. We have also raised the possibility of studying a possible improvement of the renaming of subformulae in the normal form process, which could, if successful, reduce the search space of queries. It is also important to create a method for automatically generating clause sets of known satisfiability, as the lack of available test cases can obstruct progress on future implementations. Finally, the development of a reasoner capable of working with description logics syntax would provide a useful method for studying the practical use of $\mathcal{E}$-combinations on various domains.

Bibliography

[1] P. Balsiger, A. Heuerding, and S. Schwendimann. A benchmark method for the propositional modal logics K, KT, S4. *Journal of Automated Reasoning*, 24, 01 2000. 30

[2] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, New York, NY, USA, 2001. 4, 5, 12

[3] A. Borgida and L. Serafini. Distributed description logics: Assimilating information from peer sources. *J. Data Semantics*, 1:153–184, 01 2003. 1

[4] B. Cuenca Grau, B. Parsia, and E. Sirin. Combining OWL ontologies using $\mathcal{E}$-connections. *Web Semant.*, 4(1):40–59, Jan. 2006. 1

[5] L. de Moura Amaral. $\mathcal{E}$-KSP. Available at `www.cic.unb.br/~nalon/#software`, 2019. 27

[6] M. Fisher. A resolution method for temporal logic. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'91, pages 99–104, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. 18

[7] D. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-Dimensional Modal Logics: Theory and Applications*. Elsevier, 1 edition, 2003. 2, 11, 12

[8] D. M. Gabbay. Fibred semantics and the weaving of logics. Part 1: Modal and intuitionistic logics. *Journal of Symbolic Logic*, 61(4):1057–1120, 1996. 1

[9] O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev. $\mathcal{E}$-connections of abstract description systems. *Artificial Intelligence*, 156(1):1 – 73, 2004. 1, 2

[10] A. Leitsch. *The Resolution Calculus*. Springer-Verlag New York, Inc., New York, NY, USA, 1997. 9

[11] C.-J. Liau. Belief, information acquisition, and trust in multi-agent systems—a modal logic formulation. *Artificial Intelligence*, 149(1):31 – 60, 2003. 1

[12] S. Merz. Model checking: A tutorial overview. In *Proceedings of the 4th Summer School on Modeling and Verification of Parallel Processes*, MOVEP '00, pages 3–38, London, UK, UK, 2001. Springer-Verlag. 1

[13] C. Nalon. Lógica computacional 1. Notas de Aula, 2015. 9, 10

[14] C. Nalon, U. Hustadt, and C. Dixon. A modal-layered resolution calculus for K. In H. de Nivelle, editor, *Automated Reasoning with Analytic Tableaux and Related Methods*, pages 185–200, Cham, 2015. Springer International Publishing. 27

[15] C. Nalon, U. Hustadt, and C. Dixon. $\mathcal{K}_S$P: A resolution-based prover for multimodal k. In *Automated Reasoning: 8th International Joint Conference, IJCAR 2016, Coimbra, Portugal, June 27 – July 2, 2016, Proceedings*, pages 406–415, Cham, 2016. Springer International Publishing. 27

[16] C. Nalon and O. Kutz. Towards resolution-based reasoning for connected logics. *Electron. Notes Theor. Comput. Sci.*, 305:85–102, July 2014. 2, 5, 18, 22, 23, 57

[17] J. A. Robinson. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, Jan. 1965. 9, 10, 22

[18] K. Segerberg. Two-dimensional modal logic. *Journal of Philosophical Logic*, 2(1):77–96, 1973. 1

[19] R. H. Thomason. Combinations of tense and modality. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic: Volume II: Extensions of Classical Logic*, pages 135–165. Springer Netherlands, Dordrecht, 1984. 1

Appendix A

Experimental results

A.1 Running time of tests

In this section, we include the obtained satisfiability results, together with their execution time, during the testing on the exemples provided by the K$_S$P and **LWB** bases. Any input file not listed in the **LWB** tables exceeded the timeout of 20 minutes.

File name	Output	Time (ms)
ex1.p.in	Unsatisfiable	22
ex2.p.in	Unsatisfiable	27
ex23.p.in	Unsatisfiable	20
ex3.p.in	Unsatisfiable	5
ex30.p.in	Unsatisfiable	20
ex31.p.in	Unsatisfiable	22
ex33.p.in	Unsatisfiable	28
ex34.p.in	Unsatisfiable	31
ex35.p.in	Unsatisfiable	25
ex4.p.in	Unsatisfiable	6
ex5.p.in	Unsatisfiable	6
ex6.p.in	Unsatisfiable	7
ex7.p.in	Unsatisfiable	87
ex8.p.in	Unsatisfiable	7
ex9.p.in	Unsatisfiable	20

File name	Output	Time (ms)
ex1.n.in	Satisfiable	107
ex2.n.in	Satisfiable	36
ex3.n.in	Satisfiable	8
ex32.n.in	Satisfiable	5
ex4.n.in	Satisfiable	7
ex5.n.in	Satisfiable	8
ex6.n.in	Satisfiable	6
ex7.n.in	Satisfiable	102
ex8.n.in	Satisfiable	7
ex9.n.in	Satisfiable	20

Table A.1: Running times and satisfiability for the adapted K$_S$P examples

File name	Output	Time (s)
k_branch_p.01	Unsatisfiable	2.335
k_lin_p.01	Unsatisfiable	0.010
k_lin_p.02	Unsatisfiable	0.006
k_lin_p.03	Unsatisfiable	0.006
k_lin_p.04	Unsatisfiable	0.007
k_lin_p.05	Unsatisfiable	0.006
k_lin_p.06	Unsatisfiable	0.007
k_lin_p.07	Unsatisfiable	0.009
k_lin_p.08	Unsatisfiable	0.008
k_lin_p.09	Unsatisfiable	0.008
k_lin_p.10	Unsatisfiable	0.006
k_lin_p.11	Unsatisfiable	0.011
k_lin_p.12	Unsatisfiable	0.012
k_lin_p.13	Unsatisfiable	0.009
k_lin_p.14	Unsatisfiable	0.014
k_lin_p.15	Unsatisfiable	0.012
k_lin_p.16	Unsatisfiable	0.011
k_lin_p.17	Unsatisfiable	0.006
k_lin_p.18	Unsatisfiable	0.009
k_lin_p.19	Unsatisfiable	0.009
k_lin_p.20	Unsatisfiable	0.009
k_lin_p.21	Unsatisfiable	0.008
k_path_p.01	Unsatisfiable	0.004
k_path_p.02	Unsatisfiable	108.759
k_ph_p.01	Unsatisfiable	0.006
k_ph_p.02	Unsatisfiable	0.1
k_ph_p.03	Unsatisfiable	153.494
k_poly_p.01	Unsatisfiable	1.262

File name	Output	Time (s)
k_branch_n.01	Satisfiable	7.536
k_d4_n.01	Satisfiable	2.406
k_d4_n.02	Satisfiable	22.401
k_d4_n.03	Satisfiable	564.759
k_d4_p.01	Satisfiable	7.237
k_d4_p.02	Satisfiable	338.388
k_d4_p.03	Satisfiable	21.136
k_d4_p.04	Satisfiable	231.792
k_d4_p.05	Satisfiable	251.697
k_grz_p.02	Satisfiable	15.548
k_grz_p.03	Satisfiable	24.592
k_grz_p.04	Satisfiable	48.819
k_grz_p.05	Satisfiable	104.788
k_grz_p.06	Satisfiable	285.236
k_grz_p.07	Satisfiable	792.585
k_lin_n.01	Satisfiable	0.008
k_lin_n.02	Satisfiable	16.671
k_path_n.01	Satisfiable	87.401
k_ph_n.01	Satisfiable	0.006
k_ph_n.02	Satisfiable	0.055
k_ph_n.03	Satisfiable	21.946
k_poly_n.01	Satisfiable	12.062
k_poly_n.02	Satisfiable	862.963
k_t4p_n.01	Satisfiable	258.285
k_t4p_n.02	Satisfiable	172.371
k_t4p_p.01	Satisfiable	0.720
k_t4p_p.02	Satisfiable	3.414
k_t4p_p.03	Satisfiable	8.771
k_t4p_p.04	Satisfiable	29.996
k_t4p_p.05	Satisfiable	82.099
k_t4p_p.06	Satisfiable	262.159
k_t4p_p.07	Satisfiable	730.003

Table A.2: Running times and satisfiability for the adapted **LWB**