



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Combined Proof Methods for Multimodal Logic

Daniella Angelos

Dissertação apresentada como requisito parcial para
qualificação do Mestrado em Informática

Orientadora
Prof.a Dr.a Cláudia Nalon

Brasília
2017



Universidade de Brasília

Instituto de Ciências Exatas
Departamento de Ciência da Computação

Combined Proof Methods for Multimodal Logic

Daniella Angelos

Dissertação apresentada como requisito parcial para
qualificação do Mestrado em Informática

Prof.a Dr.a Cláudia Nalon (Orientadora)
CIC/UnB

Prof. Dr. Dr.

Prof. Dr. Bruno Luigi Macchiavello Espinoza
Coordenador do Programa de Pós-graduação em Informática

Brasília, de de 2017

Abstract

Keywords: modal logics, resolution, sat-solvers

Contents

1	Introduction	1
2	Modal Logics	2
2.1	Syntax	3
2.2	Semantics	5
2.3	Normal Form	8
3	Modal-Layered Resolution	9
3.1	Clausal Resolution	9
4	Satisfiability Solvers	10
4.1	The DPLL Procedure	10
4.2	Conflict Driven Clause Learning	10
4.3	MiniSat	10
	Bibliography	12

List of Figures

2.1 Example of a Kripke model for K_n	6
2.2 Models that satisfy φ of Example 2	7

Chapter 1

Introduction

Chapter 2

Modal Logics

This chapter introduces K_n , a *propositional modal logic language*, semantically determined by an account of necessity and possibility.

A propositional modal language is the well known propositional language augmented by a collection of *modal operators*. In classical logic, propositions or sentences are either evaluated to true or false, in any model. Propositional logic and predicate logic, for instance, do not allow for any further possibilities. However, in natural language, we often distinguish between various modalities of truth, such as *necessarily* true, *known to be* true, *believed to be* true or yet true *in some future*, for example. Therefore, one may think that classical logics lacks expressivity in this sense.

Modal logic adds operators to express one or more of these different modes of truth. Different modalities define different languages. The key concept behind these operators is that they allow us to reason over relations among contexts or interpretations, an abstraction that here we think as *possible worlds*. The purpose of the modal operators is to permit the information that holds at other worlds to be examined — but, crucially, only at worlds visible from the current one via an accessibility relation [2]. Then, evaluation of a modal formula depends on a set of possible worlds and the accessibility relations defined for these worlds. It is possible to define several accessibility relations between worlds, and different modal logics are defined by different relations.

The modal language which is the focus of this work is the extension of the classical propositional logic that adds the unary operators: \Box_a and \Diamond_a , whose reading are “is necessary by the agent a ” and “is possible by the agent a ”, respectively. This language, known as K_n , is characterized by the schema $\Box_a(\varphi \Rightarrow \psi) \Rightarrow (\Box_a\varphi \Rightarrow \Box_a\psi)$ (axiom K), where $a \in \mathcal{A} = \{1, \dots, n\}$ and φ, ψ are well-formed formulae. The addition of other axioms defines different systems of modal logics and it imposes restrictions on the class of models where formulae are valid [4].

Worlds and their accessibility relations define a structure known as *Kripke model*.

The satisfiability and validity of a formula depend on this structure. For example, given a Kripke model that contains a set of possible worlds, a binary relation of accessibility between worlds and a valuation function that maps in which worlds a proposition symbol holds, we say that a formula $\Box p$ is satisfiable at some world w of this model, if the valuation function establishes that p is true at all worlds accessible from w .

It is now time to formally define the modal language we will be working with. The syntax and semantics of K_n are showed in Sections 2.1 and 2.2, respectively, and the definitions presented in these two sections are adaptations from [10].

2.1 Syntax

The language of K_n is equivalent to its set of *well-formed formulae*, denoted by WFF_{K_n} , which is constructed from a denumerable set of *propositional symbols* $\mathcal{P} = \{p, q, r, \dots\}$, the negation symbol \neg , the disjunction symbol \vee and the modal connectives \Box_a , that express the notion of necessity, for each index a in a finite, non-empty fixed set of labels $\mathcal{A} = \{1, \dots, n\}, n \in \mathbb{N}$.

Definition 1 The set of well-formed formulae, WFF_{K_n} , is the least set such that:

1. $p \in WFF_{K_n}$, for all $p \in \mathcal{P}$
2. if $\varphi, \psi \in WFF_{K_n}$, then so are $\neg\varphi, (\varphi \vee \psi)$ and $\Box_a \varphi$, for each $a \in \mathcal{A}$

Just as the familiar first-order existential and universal quantifiers are duals to each other, that is, $\forall x \varphi \Leftrightarrow \neg \exists x \neg \varphi$, we have the dual connectives \Diamond for necessity, which express possibility, and they are defined by $\Diamond \varphi \stackrel{\text{def}}{=} \neg \Box_a \neg \varphi$, for each $a \in \mathcal{A}$. Other logic operators may be used as abbreviations. In this work, we consider the usual ones:

- $\varphi \wedge \psi \stackrel{\text{def}}{=} \neg(\neg\varphi \vee \neg\psi)$ (conjunction)
- $\varphi \Rightarrow \psi \stackrel{\text{def}}{=} \neg\varphi \vee \psi$ (implication)
- $\varphi \Leftrightarrow \psi \stackrel{\text{def}}{=} (\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi)$ (equivalence)
- **false** $\stackrel{\text{def}}{=} \varphi \wedge \neg\varphi$ (*falsum*)
- **true** $\stackrel{\text{def}}{=} \neg\text{false}$ (*verum*)

Parentheses may be omitted if the reading is not ambiguous. When $n = 1$, we often omit the index in the modal operators, i.e., we just write $\Box\varphi$ (or ‘box’ φ) and $\Diamond\varphi$ (or ‘diamond’ φ), for a well-formed formula φ .

We define as *literal* a propositional symbol $p \in \mathcal{P}$ or its negation $\neg p$, and denote by \mathcal{L} the set of all literals. A *modal literal* is a formula of the form $\boxed{a}l$ or $\Diamond l$, with $l \in \mathcal{L}$ and $a \in \mathcal{A}$.

The following function definitions are based on formulae' syntax. The *modal depth* of a formula is recursively defined as follows:

Definition 2 We define $mdepth : \mathbf{WFF}_{K_n} \longrightarrow \mathbb{N}$ to represent the maximal number of nesting modal operators in a formula. Inductively:

1. $mdepth(p) = 0$
2. $mdepth(\neg\varphi) = mdepth(\varphi)$
3. $mdepth(\varphi \vee \psi) = \max\{mdepth(\varphi), mdepth(\psi)\}$
4. $mdepth(\boxed{a}\varphi) = mdepth(\varphi) + 1$

With $p \in \mathcal{P}$ and $\varphi, \psi \in \mathbf{WFF}_{K_n}$.

For instance, if $\varphi = \boxed{a}\Diamond p$ then $mdepth(\varphi) = 2$ but $mdepth(p) = 0$.

The *modal level* of a formula (or a subformula) is given relative to its position in the *annotated syntactic tree*.

Definition 3 Let Σ be the alphabet $\{1, 2, .\}$ and Σ^* the set of all finite sequences over Σ . Denote by ε the empty sequence. We define $\tau : \mathbf{WFF}_{K_n} \times \Sigma^* \times \mathbb{N} \longrightarrow \mathcal{P}(\mathbf{WFF}_{K_n} \times \Sigma^* \times \mathbb{N})$ as the partial function inductively defined as follows:

1. $\tau(p, \lambda, ml) = \{(p, \lambda, ml)\}$
2. $\tau(\neg\varphi, \lambda, ml) = \{(\neg\varphi, \lambda, ml)\} \cup \tau(\varphi, \lambda.1, ml)$
3. $\tau(\boxed{a}\varphi, \lambda, ml) = \{(\boxed{a}\varphi, \lambda, ml)\} \cup \tau(\varphi, \lambda.1, ml + 1)$
4. $\tau(\varphi \vee \psi, \lambda, ml) = \{(\varphi \vee \psi, \lambda, ml)\} \cup \tau(\varphi, \lambda.1, ml) \cup \tau(\psi, \lambda.2, ml)$

With $p \in \mathcal{P}$, $\lambda \in \Sigma^*$, $ml \in \mathbb{N}$ and $\varphi, \psi \in \mathbf{WFF}_{K_n}$.

The function τ applied to $(\varphi, \varepsilon, 0)$ returns the annotated syntactic tree for φ , where each node is uniquely identified by a subformula, its position in the tree (or path order) and its modal level. For instance, p occurs twice in the formula $\boxed{a}\Diamond(p \wedge \boxed{a}p)$, at the position 1.1.1, with modal level 2, and again at position 1.1.2.1, with modal level 3.

Definition 4 We define $mlevel : \mathbf{WFF}_{K_n} \times \mathbf{WFF}_{K_n} \times \Sigma^* \longrightarrow \mathbb{N}$, to represent the

maximal number of modal operators in which scope a subformula occurs. Let φ be a formula and let $\tau(\varphi, \varepsilon, 0)$ be its annotated syntactic tree. If $(\varphi', \lambda, ml) \in \tau(\varphi, \varepsilon, 0)$ then $mlevel(\varphi, \varphi', \lambda) = ml$.

2.2 Semantics

The semantics of \mathbf{K}_n is presented in terms of Kripke structures.

Definition 5 A Kripke model for \mathcal{P} and $\mathcal{A} = \{1, \dots, n\}$ is given by the tuple

$$\mathcal{M} = (W, w_0, R_1, \dots, R_n, \pi) \quad (2.1)$$

where W is a non-empty set of possible worlds with a distinguished world w_0 , the root of \mathcal{M} ; each R_a , $a \in \mathcal{A}$, is a binary relation on W , that is, $R_a \subseteq W \times W$, and $\pi : W \times \mathcal{P} \longrightarrow \{\text{false}, \text{true}\}$ is the valuation function that associates to each world $w \in W$ a truth-assignment to propositional symbols.

We denote $R_a wv$ to mean that v is accessible from w through the accessibility relation R_a , that is $(w, v) \in R_a$, and $R_a^* wv$, to mean that v is reachable from w through a finite number of steps, that is, exists a sequence (w_1, \dots, w_k) of worlds such that $R_a w_i w_{i+1}, \forall 1 \leq i < k$, where $w_1 = w$ and $w_k = v$, with $a \in \mathcal{A}$, $w, v, w_i \in W$ and $k \in \mathbb{N}$. Note that R_a^* is the *transitive closure* of R_a , the least transitive set that contains all elements of R_a . Here we also make use of the *transitive and reflexive closure*, denoted by R_a^+ , the least transitive and reflexive set that contains all elements of R_a .

Satisfiability and *validity* of a formula is defined in terms of the *satisfiability relation*.

Definition 6 Let $\mathcal{M} = (W, w_0, R_1, \dots, R_n, \pi)$ be a Kripke model, $w \in W$ and $\varphi, \psi \in \mathbf{WFF}_{\mathbf{K}_n}$. The *satisfiability relation*, denoted by $\langle \mathcal{M}, w \rangle \models \varphi$, between a world w and a formula φ , is inductively defined by:

1. $\langle \mathcal{M}, w \rangle \models p$ if, and only if, $\pi(w, p) = \mathbf{true}$, for all $p \in \mathcal{P}$;
2. $\langle \mathcal{M}, w \rangle \models \neg\varphi$ if, and only if, $\langle \mathcal{M}, w \rangle \not\models \varphi$;
3. $\langle \mathcal{M}, w \rangle \models \varphi \vee \psi$ if, and only if, $\langle \mathcal{M}, w \rangle \models \varphi$ or $\langle \mathcal{M}, w \rangle \models \psi$;
4. $\langle \mathcal{M}, w \rangle \models \boxed{a}\varphi$ if, and only if, for all $t \in W$, $(w, t) \in R_a$ implies $\langle \mathcal{M}, t \rangle \models \varphi$

Satisfiability is defined with respect to the root of a model. A formula $\varphi \in \mathbf{WFF}_{\mathbf{K}_n}$ is said to be *satisfiable* if there exists a Kripke model $\mathcal{M} = (W, w_0, R_1, \dots, R_n, \pi)$ such

that $\langle \mathcal{M}, w_0 \rangle \models \varphi$. In this case we simply write $\mathcal{M} \models \varphi$ to mean that \mathcal{M} satisfies φ at w_0 . A formula is said to be *valid* if it is satisfiable in all models. We say that a set \mathcal{F} is satisfiable if every $\varphi \in \mathcal{F}$ is satisfiable. Validity of sets is defined analogously.

The satisfiability problem for K_n corresponds to determining the existence of a model in which a formula is satisfied. This problem is proven to be PSPACE-complete [12].

Example 1 Consider the model illustrated in Figure 2.1, let's call it \mathcal{M} . Take $\mathcal{M} = (W, w_0, R, \pi)$, for $\mathcal{P} = \{p\}$ and $\mathcal{A} = \{1\}$, where

- (i) $W = \{w_0, w_1, w_2\}$
- (ii) $R = \{(w_1, w_1), (w_2, w_2), (w_0, w_1), (w_0, w_2)\}$
- (iii) $\pi(w, p) = \begin{cases} \text{true} & \text{if } w = w_0 \\ \text{false} & \text{otherwise} \end{cases}$

Note that both p and $\Box \neg p$ are satisfied in \mathcal{M} . This is a rather simple example to illustrate that, even though some sentence evaluates to true in the current context, one can see the same sentence occurring with the opposite valuation through an accessibility relation. This kind of reasoning is not possible in classical logic. Other examples of formulae satisfied by this model: $p \wedge \Diamond \neg p$, $\Box \Box \neg p$ and $\Box \Box \Box \neg p$.

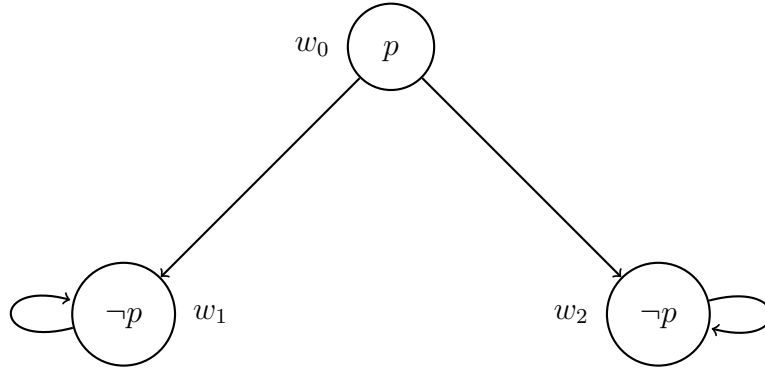


Figure 2.1: Example of a Kripke model for K_n

Example 2 (*Tree-like model*) Consider the formula $\varphi = \Box(p \Rightarrow \Diamond p)$. The Figure 2.2 contains examples of models that satisfy φ . Note that the model from the figure 2.2b has a graphical representation similar to a tree. Finite trees are ubiquitous in computer science, they are used to represent knowledge or data from the most diverse fields. As trees play such an important role, we will take this opportunity to define them, and next, we will show some interesting results concerning our modal language.

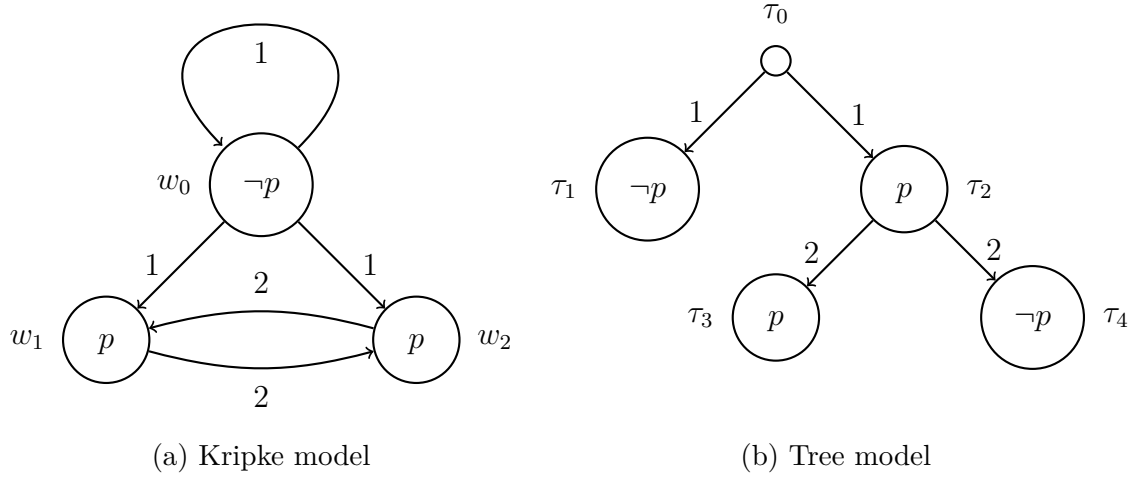


Figure 2.2: Models that satisfy φ of Example 2

By a tree \mathcal{T} we mean a relational structure (T, S) where T is a set of nodes and S is a binary relation among these nodes. T contains a unique $r_0 \in T$ (called the *root*) such that all other nodes in T are reachable from r_0 , that is, $\forall t \in T$ we have S^*r_0t , besides that, every element of T , distinct from r_0 , has a unique S -predecessor, and the transitive and reflexive closure S^+ is acyclic, that is, $\forall t \in T, \neg S^+tt$ [1].

A *tree model* is a Kripke model (W, r_0, R, π) , with $\mathcal{A} = \{1\}$, where (W, R) is a tree and r_0 is its root. A *tree-like model* for \mathbf{K}_n is a model $(W, r_0, R_1, \dots, R_n, \pi)$, with $\mathcal{A} = \{1, \dots, n\}$, such that $(W, \cup_{i \in \mathcal{A}} R_i)$ is a tree, with r_0 as the root.

In a tree-like model, we can define the *depth* of worlds in W .

Definition 7 Let $\mathcal{M} = (W, r_0, R_1, \dots, R_n, \pi)$ be a tree-like model for \mathbf{K}_n . We define the *depth* : $W \rightarrow \mathbb{N}$ of a world $w \in W$, as the length of the path from r_0 to w through the union of the relations in \mathcal{M} . We sometimes say *depth* of \mathcal{M} to mean the largest path from the root to a world in W .

Theorem 2.2.1. Let $\varphi \in \text{WFF}_{\mathbf{K}_n}$ be a formula and $\mathcal{M} = (W, w_0, R_1, \dots, R_n, \pi)$ be a model. Then $\mathcal{M} \models \varphi$ if and only if there is a tree-like model \mathcal{M}' such that $\mathcal{M}' \models \varphi$. Moreover, \mathcal{M}' is finite and its depth is bounded by $\text{mdepth}(\varphi)$.

Proof. □

Theorem 2.2.2. Let $\varphi, \varphi' \in \text{WFF}_{\mathbf{K}_n}$ and $\mathcal{M} = (W, w_0, R_1, \dots, R_n, \pi)$ be a tree-like model such that $\mathcal{M} \models \varphi$. If $(\varphi', \lambda', ml) \in \tau(\varphi, \varepsilon, 0)$ and φ' is satisfied in \mathcal{M} , then there is $w \in W$, with $\text{depth}(w) = ml$, such that $\langle \mathcal{M}, w \rangle \models \varphi'$. Moreover, the subtree rooted at w has height equals to $\text{mdepth}(\varphi')$.

Proof. □

The *global satisfiability* problem for a modal logic is equivalent to the satisfiability problem of the logic obtained by adding the universal modality, \Box , to the original language [9]. Let K_n^* be the logic obtained by adding \Box to K_n . Let $\mathcal{M} = (W, w_0, R_1, \dots, R_n, \pi)$ be a tree-like model for K_n . A model \mathcal{M}^* for K_n^* is the pair (\mathcal{M}, R_*) , where $R_* = W \times W$. A formula $\Box\varphi$ is satisfied at the world $w \in W$, in the model \mathcal{M}^* , written $\langle \mathcal{M}^*, w \rangle \models \Box\varphi$, if, and only if, for all $w' \in W$, we have that $\langle \mathcal{M}^*, w' \rangle \models \varphi$. In this sense, let $\varphi \in \text{WFF}_{K_n}$ be a formula, we say that φ is globally satisfiable in a model \mathcal{M} , denoted $\mathcal{M} \models_G \varphi$, if, and only if, $\mathcal{M}^* \models \Box\varphi$.

2.3 Normal Form

Normal forms can provide elegant and constructive proofs of many standard results [7]. This happens because once you translated a set of formulae into a normal form, you have all your formulae in a specific structure and possibly less operators to work with, which may implicate into a smaller number of rules for a proof system. One can even add information into formulae in a specific normal form. That is the case of the normal form we use in this work.

Formulae in K_n can be transformed into a layered normal form called *Separated Normal Form with Modal Levels*, denoted by SNF_{ml} [10]. A formula in SNF_{ml} is a conjunction of *clauses* labelled by the modal level in which they occur.

We write $ml : \varphi$ to denote that φ occurs at modal level $ml \in \mathbb{N} \cup \{*\}$. By $* : \varphi$ we mean that φ is true at all modal levels. Formally, let $\text{WFF}_{K_n}^{ml}$ denote the set of formulae with the modal level annotation, $ml : \varphi$ such that $ml \in \mathbb{N} \cup \{*\}$ and $\varphi \in \text{WFF}_{K_n}$. Let $\mathcal{M}^* = (W, w_0, R_1, \dots, R_n, R_*, \pi)$ be a model and take $\varphi \in \text{WFF}_{K_n}$.

Definition 8 Satisfiability of labelled formulae is given by:

1. $\mathcal{M}^* \models ml : \varphi$ if, and only if, for all worlds $w \in W$ such that $\text{depth}(w) = ml$, we have $\langle \mathcal{M}^*, w \rangle \models \varphi$
2. $\mathcal{M}^* \models * : \varphi$ if, and only if, $\mathcal{M}^* \models \Box\varphi$

Chapter 3

Modal-Layered Resolution

Resolution appeared in the early 1960s through investigations on performance improvements of refutational procedures based on *Herbrand Theorem*. In particular, Prawitz' studies on such procedures brought back the idea of unification. J. A. Robinson incorporated the concept of unification on a refutation method, creating what was later known as resolution [3].

The standard rules for resolution systems take two or more premises with literals or modal literals that are contradictory, and generate a resolvent. Most of these systems work exclusively with clauses in a specific normal form. Resolution systems are, in general, refutational systems, that is, to show that a formula φ is valid, $\neg\varphi$ is translated into a normal form. The inference rules are applied until either no new resolvents can be generated or a contradiction is obtained. The contradiction implies that $\neg\varphi$ is unsatisfiable and hence, that φ is valid.

3.1 Clausal Resolution

Clausal resolution was proposed as a proof method for classical logic by Robinson in 1965 [11], and was claimed to be suitable to be performed by computer, as it has only one inference rule that is applied many times. Nonclausal proof methods, in general, require a larger number of rules, making implementation more difficult. Clausal resolution is a simple and adaptable proof method for classical logics and, since it was proposed, a bank of research into heuristics and strategies has been growing.

Chapter 4

Satisfiability Solvers

The problem of determining whether a boolean formula is satisfiable has the historical honor of being the first problem ever shown to be NP-Complete [5]. Great theoretical and practical efforts have been directed in improving the efficiency of solvers for this problem, known as *Boolean Satisfiability Solvers*, or just *SAT solvers*. Despite the worst-case exponential run time of all the algorithms known, satisfiability solvers are increasingly leaving their mark as a general purpose tool in the most diverse areas [8]. In essence, SAT solvers provide a generic combinatorial reasoning and search platform.

In the context of SAT solvers, the underlying representational formalism is propositional logic. We are interested in propositional formulae in *Conjunctive Normal Form* (CNF): F is in CNF if it is a conjunction of *clauses*, where each clause is a disjunction of *literals*. For example, $F = (p \vee \neg q) \wedge (\neg p \vee r \vee s) \wedge (q \vee r)$ is a CNF formula with four variables and three clauses.

Therefore, the *Boolean Satisfiability Problem* (SAT) can be expressed as: Given a CNF formula F , does F have a satisfying assignment? One can be interested not only in the answer of this decision problem, but also in finding an actual satisfying assignment when it there exists one. All practical SAT solvers do produce such assignment [6].

4.1 The DPLL Procedure

4.2 Conflict Driven Clause Learning

4.3 MiniSat

Algorithm 1: DPLL-recursive(F, ρ)

Input : A CNF formula F and an initially empty partial assignment ρ

Output: UNSAT, or an assignment satisfying F

```
1  $(F, \rho) \leftarrow \text{UnitPropagate}(F, \rho)$ 
2 if  $F$  contains the empty clause then
3   | return UNSAT
4 end
5 if  $F$  has no clauses left then
6   | Output  $\rho$ 
7   | return SAT
8 end
9  $l \leftarrow$  a literal not assigned by  $\rho$ 
10 if  $\text{DPLL-recursive}(F|_l, \rho \cup \{l\}) = \text{SAT}$  then
11   | return SAT
12 end
13 return  $\text{DPLL-recursive}(F|_{\neg l}, \rho \cup \{\neg l\})$ 
```

Algorithm 2: CDCL(F, ν)

Input :

Output:

```
1 if  $\text{UnitPropagate}(F, \nu) == \text{CONFLICT}$  then
2   | return UNSAT
3 end
4  $dl \leftarrow 0$ 
5 while not  $\text{AllVariablesAssigned}(F, \nu)$  do
6   |  $(x, \nu) \leftarrow \text{PickBranchingVariable}(F, \nu)$ 
7 end
8 return SAT
```

Bibliography

- [1] C. Areces, R. Gennari, J. Heguiabehere, and M. De Rijke. Tree-based heuristics in modal theorem proving. In *Proceedings of the 14th European Conference on Artificial Intelligence*, pages 199–203. IOS Press, 2000. 7
- [2] P. Blackburn, M. De Rijke, and Y. Venema. *Modal Logic: Graph. Darst*, volume 53. Cambridge University Press, 2002. 2
- [3] M. A. Casanova. *Programação em lógica e a linguagem Prolog*. E. Blucher, 1987. 9
- [4] B. F. Chellas. *Modal logic — an introduction*. Press Syndicate of the University of Cambridge, London, 1980. 2
- [5] S. A. Cook. The complexity of theorem proving procedures. In *stoc71*, pages 151–158, 1971. 10
- [6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, 3rd Edition*. MIT Press, 2009. 10
- [7] K. Fine. Normal forms in modal logic. *Notre Dame J. Formal Logic*, 16(2):229–237, 04 1975. 8
- [8] C. P. Gomes, H. Kautz, A. Sabharwal, and B. Selman. Satisfiability solvers. *Foundations of Artificial Intelligence*, 3:89–134, 2008. 10
- [9] V. Goranko and S. Passy. Using the universal modality: gains and questions. *Journal of Logic and Computation*, 2(1):5–30, 1992. 8
- [10] C. Nalon and C. Dixon. Clausal resolution for normal modal logics. *J. Algorithms*, 62(3-4):117–134, 2007. 3, 8
- [11] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, Jan. 1965. 9
- [12] E. Spaan. *Complexity of Modal Logics*. PhD thesis, University of Amsterdam, 1993. 6