Efficient Theorem-Proving for Modal Logics

Cláudia Nalon

Department of Computer Science, University of Brasília

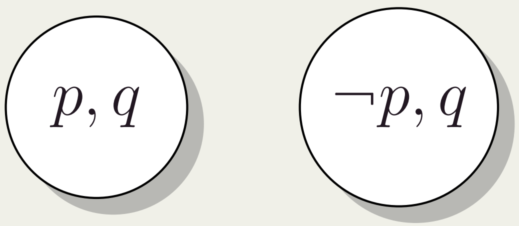
Joint work with Clare Dixon (Manchester), Ullrich Hustadt (Liverpool), and Fabio Papacchini (Lancaster at Leipzig)

Introduction

Motivation

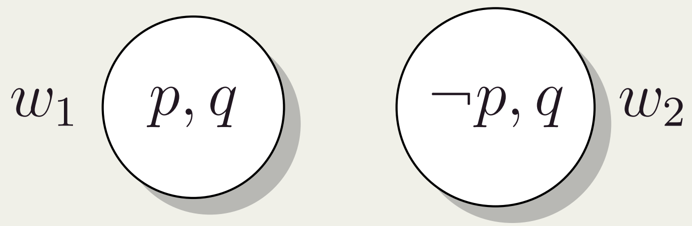
- Modal logics have been used in Computer Science to represent properties of complex systems: temporal, epistemic, obligations, choice, actions, and so on.
- Given a representation of a computational system in a logical language, we also want to reason about the system and their properties.
- There are different proof methods we could use:
  - Some modal languages can be translated into first-order and we could then use readily available automated reasoners.
  - Provide a proof method within the language of a particular modal logic.

- Modal logics are extensions of propositional logic with operators '$\Box$' and '$\Diamond$'.
- Evaluation of a formula depends on a set of worlds and on the accessibility relations on this set.
- Different restrictions on the accessibility relations give rise to different modal logics.



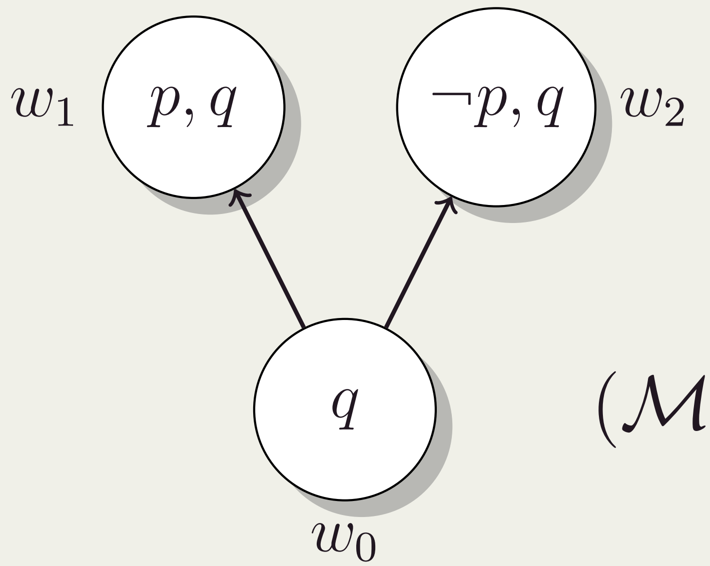
Syntax and Semantics

- Modal logics are extensions of propositional logic with operators '$\Box$' and '$\Diamond$'.
- Evaluation of a formula depends on a set of worlds and on the accessibility relations on this set.
- Different restrictions on the accessibility relations give rise to different modal logics.



Syntax and Semantics

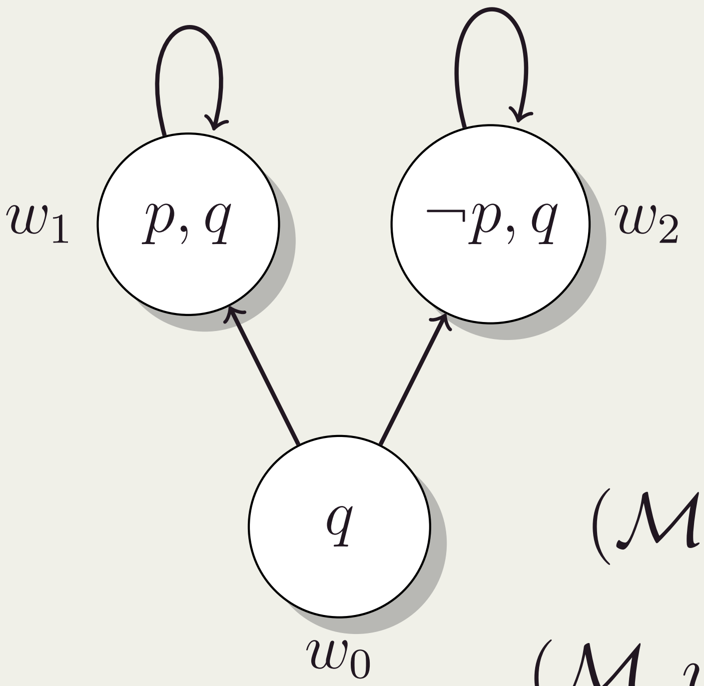
- Modal logics are extensions of propositional logic with operators '$\Box$' and '$\Diamond$'.
- Evaluation of a formula depends on a set of worlds and on the accessibility relations on this set.
- Different restrictions on the accessibility relations give rise to different modal logics.



$$(\mathcal{M}, w_0) \models \Diamond p \wedge \Diamond \neg p$$

Syntax and Semantics

- Modal logics are extensions of propositional logic with operators '$\Box$' and '$\Diamond$'.
- Evaluation of a formula depends on a set of worlds and on the accessibility relations on this set.
- Different restrictions on the accessibility relations give rise to different modal logics.

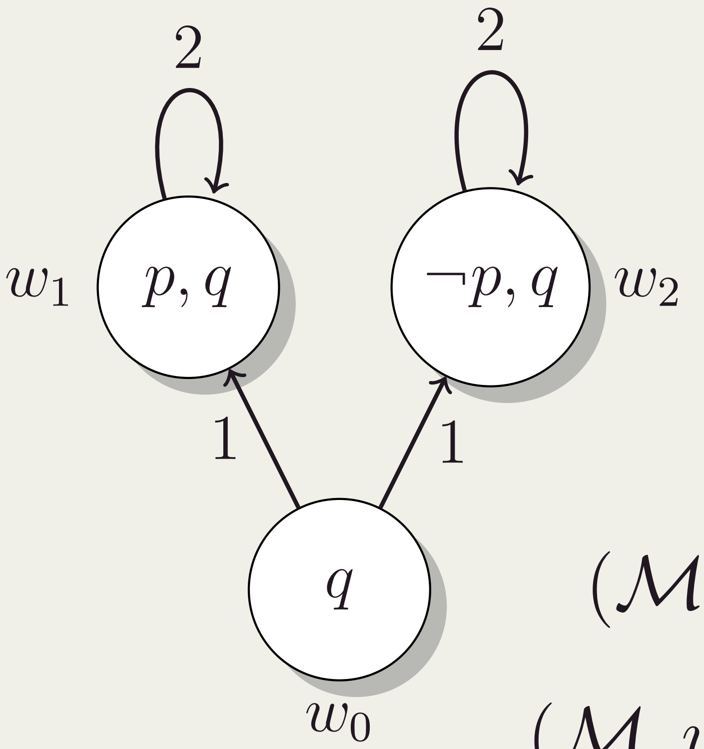


$$(\mathcal{M}, w_0) \models \Diamond p \wedge \Diamond \neg p$$

$$(\mathcal{M}, w_0) \models \Box(\Box p \vee \Box \neg p)$$

Syntax and Semantics

- Modal logics are extensions of propositional logic with operators '$\boxed{a}$' and '$\langle a \rangle$', where $a \in \mathcal{A} = \{1, \ldots, n\}$, $n \in \mathbb{N}$.
- Evaluation of a formula depends on a set of worlds and on the accessibility relations on this set.
- Different restrictions on the accessibility relations give rise to different modal logics.



$$(\mathcal{M}, w_0) \models \langle 1 \rangle p \wedge \langle 1 \rangle \neg p$$

$$(\mathcal{M}, w_0) \models \boxed{1}(\boxed{2}p \vee \boxed{2}\neg p)$$

Syntax

- The set of well-formed formulae, WFF:
  - $p \in \mathcal{P}$;
  - if $\varphi \in$ WFF, then so are $\neg\varphi$ and $\boxed{a}\varphi$, $a \in \mathcal{A} = \{1, \ldots, n\}$;
  - if φ and $\psi \in$ WFF, then $(\varphi \wedge \psi) \in$ WFF.
- Abbreviations:
  - $\text{false} \equiv p \wedge \neg p$ (for $p \in \mathcal{P}$)
  - $\text{true} \equiv \neg\text{false}$
  - $\varphi \vee \psi \equiv \neg(\neg\varphi \wedge \neg\psi)$
  - $\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$
  - $\varphi \leftrightarrow \psi \equiv (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$
  - $\langle a \rangle\varphi \equiv \neg\boxed{a}\neg\varphi$.

Semantics

- A Kripke Structure $\mathcal{M}$ for $\mathcal{P}$ and $\mathcal{A} = \{1, \ldots, n\}$ is a tuple

$$\mathcal{M} = \langle \mathcal{W}, \mathcal{R}_1, \ldots, \mathcal{R}_n, \pi \rangle,$$

  where:

  - $\mathcal{W}$ is a non-empty set;
  - For each $a \in \mathcal{A}$, $\mathcal{R}_a \subseteq \mathcal{W} \times \mathcal{W}$;
  - $\pi : \mathcal{W} \times \mathcal{P} \longrightarrow \{T, F\}$.

- The satisfiability relation $\models$ between a world $w \in \mathcal{W}$ in a Kripke structure $\mathcal{M}$ and a formula is inductively defined by:

  - $(\mathcal{M}, w) \models p$, $p \in \mathcal{P}$, iff $\pi(w, p) = T$;
  - $(\mathcal{M}, w) \models \neg\varphi$ iff $(\mathcal{M}, w) \not\models \varphi$;
  - $(\mathcal{M}, w) \models \varphi \wedge \psi$ iff $(\mathcal{M}, w) \models \varphi$ and $(\mathcal{M}, w) \models \psi$;
  - $(\mathcal{M}, w) \models \boxed{a}\varphi$ iff for all w', $w\mathcal{R}_a w'$ implies $(\mathcal{M}, w') \models \varphi$.

Reasoning Tasks

$$\mathcal{M} = \langle \mathcal{W}, \mathcal{R}_1, \ldots, \mathcal{R}_n, \pi \rangle$$

- A formula φ is locally satisfiable iff there is a model $\mathcal{M}$ and $w \in \mathcal{W}$ such that $\langle \mathcal{M}, w \rangle \models \varphi$. In this case, we say that $\mathcal{M}$ satisfies φ, denoted by $\mathcal{M} \models_L \varphi$.

- A formula φ is globally satisfiable iff there is a model $\mathcal{M}$ and for all $w \in \mathcal{W}$ we have that $\langle \mathcal{M}, w \rangle \models \varphi$. In this case, we say that $\mathcal{M}$ globally satisfies φ, denoted by $\mathcal{M} \models_G \varphi$.

- A formula φ is satisfiable under the global constraints $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ iff there is a model $\mathcal{M}$ such that $\mathcal{M} \models_G \Gamma$ and there is $w \in \mathcal{W}$ such that $\langle \mathcal{M}, w \rangle \models_L \varphi$.

Reasoning Tasks

$$\mathcal{M} = \langle \mathcal{W}, \mathcal{R}_1, \ldots, \mathcal{R}_n, \pi \rangle$$

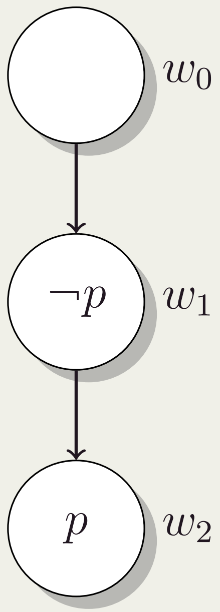
- A formula φ is locally satisfiable iff there is a model $\mathcal{M}$ and $w \in \mathcal{W}$ such that $\langle \mathcal{M}, w \rangle \models \varphi$. In this case, we say that $\mathcal{M}$ satisfies φ, denoted by $\mathcal{M} \models_L \varphi$.
  PSPACE-complete [Ladner, 1977, Halpern and Moses, 1992]

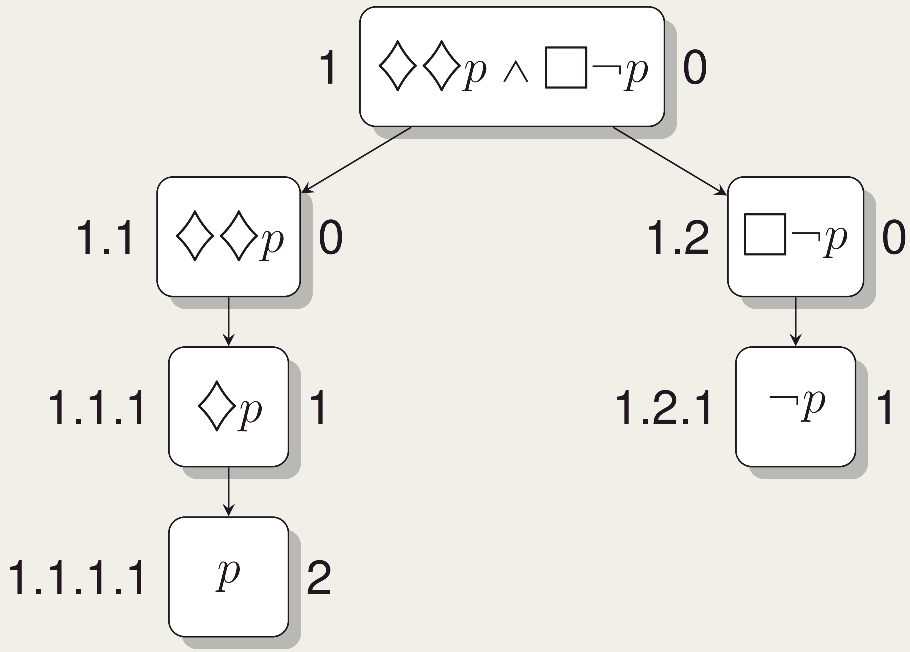
- A formula φ is globally satisfiable iff there is a model $\mathcal{M}$ and for all $w \in \mathcal{W}$ we have that $\langle \mathcal{M}, w \rangle \models \varphi$. In this case, we say that $\mathcal{M}$ globally satisfies φ, denoted by $\mathcal{M} \models_G \varphi$.
  EXPTIME-complete [Spaan, 1993]

- A formula φ is satisfiable under the global constraints $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ iff there is a model $\mathcal{M}$ such that $\mathcal{M} \models_G \Gamma$ and there is $w \in \mathcal{W}$ such that $\langle \mathcal{M}, w \rangle \models_L \varphi$.
  EXPTIME-complete [Spaan, 1993]

- Nice properties: finite, tree-like models with height bounded by the modal depth/modal level of the formula.

$$\Diamond\Diamond p \wedge \Box\neg p$$





Clausal Resolution for Propositional Logic

- There is only one inference rule:

$$[\text{RES}]\quad \frac{\begin{array}{ccr}\varphi & \vee & l\\ \psi & \vee & \neg l\end{array}}{\begin{array}{ccr}\varphi & \vee & \psi\end{array}}$$

Procedure

Let Γ_0 be a set of clauses.

$i \leftarrow 0$
repeat
    Choose c_1 and $c_2 \in \Gamma_i$ such that $l \in c_1$ and $\neg l \in c_2$
    Calculate the resolvent r
    if r is not redundant **then**
        Let $\Gamma_{i+1} \leftarrow \Gamma_i \cup \{r\}$
    end if
    $i \leftarrow i + 1$
until $\mathrm{false} \in \Gamma_i$ or $\Gamma_{i+1} = \Gamma_i$

Procedure

Let Γ_0 be a set of clauses.

$i \leftarrow 0$
repeat
    Choose c_1 and $c_2 \in \Gamma_i$ such that $l \in c_1$ and $\neg l \in c_2$
    Calculate the resolvent r
    if r is not redundant **then**
        Let $\Gamma_{i+1} \leftarrow \Gamma_i \cup \{r\}$
    end if
    $i \leftarrow i + 1$
until $\text{false} \in \Gamma_i$ or $\Gamma_{i+1} = \Gamma_i$

Procedure

Let Γ_0 be a set of clauses.

$i \leftarrow 0$
repeat
    Choose c_1 and $c_2 \in \Gamma_i$ such that $l \in c_1$ and $\neg l \in c_2$
    Calculate the resolvent r
    if r is not redundant **then**
        Let $\Gamma_{i+1} \leftarrow \Gamma_i \cup \{r\}$
    end if
    $i \leftarrow i + 1$
until $\mathrm{false} \in \Gamma_i$ or $\Gamma_{i+1} = \Gamma_i$

Procedure

Let Γ_0 be a set of clauses.

$i \leftarrow 0$
repeat
    Choose c_1 and $c_2 \in \Gamma_i$ such that $l \in c_1$ and $\neg l \in c_2$
    Calculate the resolvent r
    if r is not redundant **then**
        Let $\Gamma_{i+1} \leftarrow \Gamma_i \cup \{r\}$
    end if
    $i \leftarrow i + 1$
until $\text{false} \in \Gamma_i$ or $\Gamma_{i+1} = \Gamma_i$

CNF

- conjunctive normal form

$$\bigwedge_{i=1}^{n} \bigvee_{j=1}^{m} l_{ij}$$

Let $\varphi \in$ WFF. There is $\varphi' \in$ WFF, $\varphi' \mathrel{=}\!\!|\!\!|\!\!\models \varphi$ and φ' is in CNF.

CNF

- conjunctive normal form

$$\bigwedge_{i=1}^{n} \bigvee_{j=1}^{m} l_{ij}$$

> Let $\varphi \in$ WFF. There is $\varphi' \in$ WFF, $\varphi' \dashv\vDash \varphi$ and φ' is in CNF.

- $\varphi \rightarrow \varphi' \longmapsto \neg\varphi \vee \varphi'$ (def. implication);
- $\neg(\varphi \wedge \varphi') \longmapsto \neg\varphi \vee \neg\varphi'$ (De Morgan);
- $\neg(\varphi \vee \varphi') \longmapsto \neg\varphi \wedge \neg\varphi'$ (De Morgan);
- $\neg\neg\varphi \longmapsto \varphi$ (double negation elimination);
- $\varphi \vee (\varphi' \wedge \varphi'') \longmapsto (\varphi \vee \varphi') \wedge (\varphi \vee \varphi'')$ (distribution).

- conjunctive normal form

$$\bigwedge_{i=1}^{n} \bigvee_{j=1}^{m} l_{ij}$$

Let $\varphi \in$ WFF. There is $\varphi' \in$ WFF, $\varphi' \dashv\models \varphi$ and φ' is in CNF.

- $\varphi \to \varphi' \longmapsto \neg\varphi \vee \varphi'$ (def. implication);
- $\neg(\varphi \wedge \varphi') \longmapsto \neg\varphi \vee \neg\varphi'$ (De Morgan);
- $\neg(\varphi \vee \varphi') \longmapsto \neg\varphi \wedge \neg\varphi'$ (De Morgan);
- $\neg\neg\varphi \longmapsto \varphi$ (double negation elimination);
- $\varphi \vee (\varphi' \wedge \varphi'') \longmapsto (\varphi \vee \varphi') \wedge (\varphi \vee \varphi'')$ (distribution).

$$\text{size}((\varphi \vee \varphi') \wedge (\varphi \vee \varphi'')) = 2 \times \text{size}(\varphi) + \text{size}(\varphi' \wedge \varphi'') + 2$$

Renaming

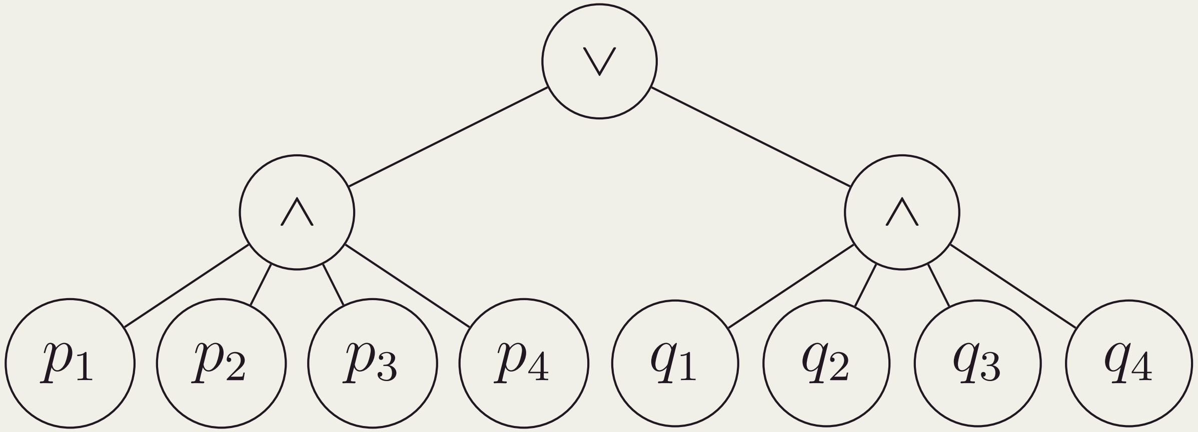
- Introduce new literals which replace complex subformulae;
- Introduce the definition clauses for those literals.

Let φ be the formula to be replaced and new_φ a fresh propositional symbol:

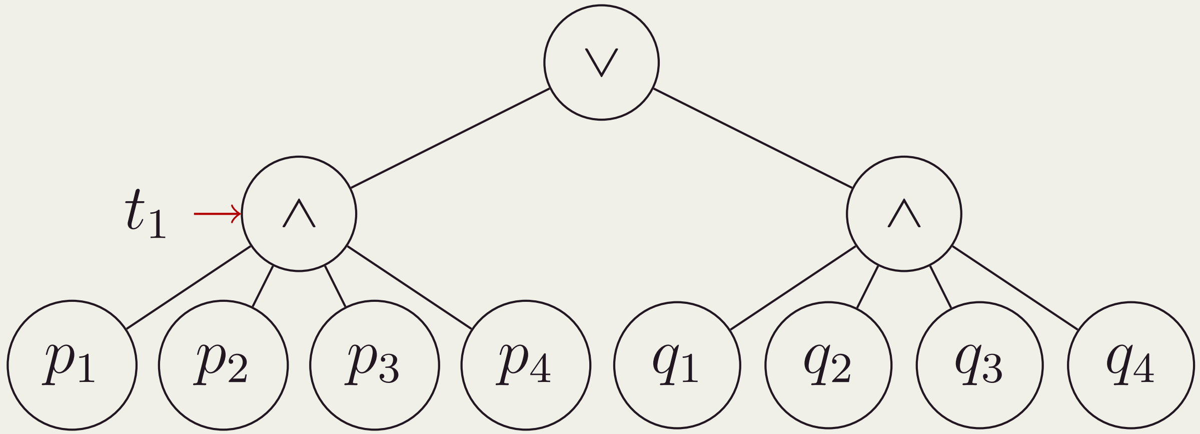
$$
\begin{array}{rcl}
Pol(\varphi) > 0 & \Rightarrow & new_\varphi \rightarrow \varphi \\
Pol(\varphi) < 0 & \Rightarrow & \varphi \rightarrow new_\varphi \\
Pol(\varphi) = 0 & \Rightarrow & new_\varphi \leftrightarrow \varphi
\end{array}
$$

[Tseitin,1968],[PG, 1986]
Let $\varphi \in$ WFF. There is $\varphi' \in$ WFF, φ' is in CNF, and φ' is satisfiable if, and only if, φ is satisfiable. Moreover, size$(\varphi') = O($size$(\varphi))$.

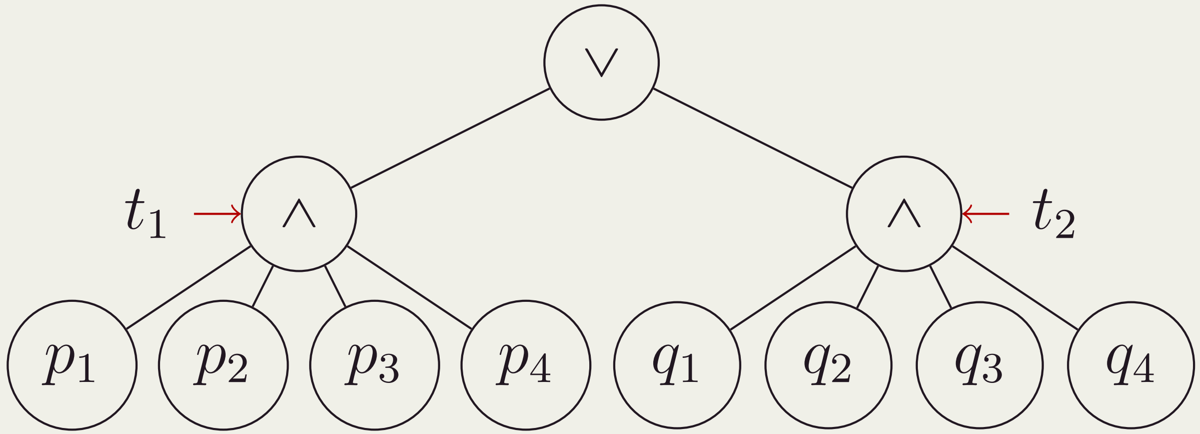
Example



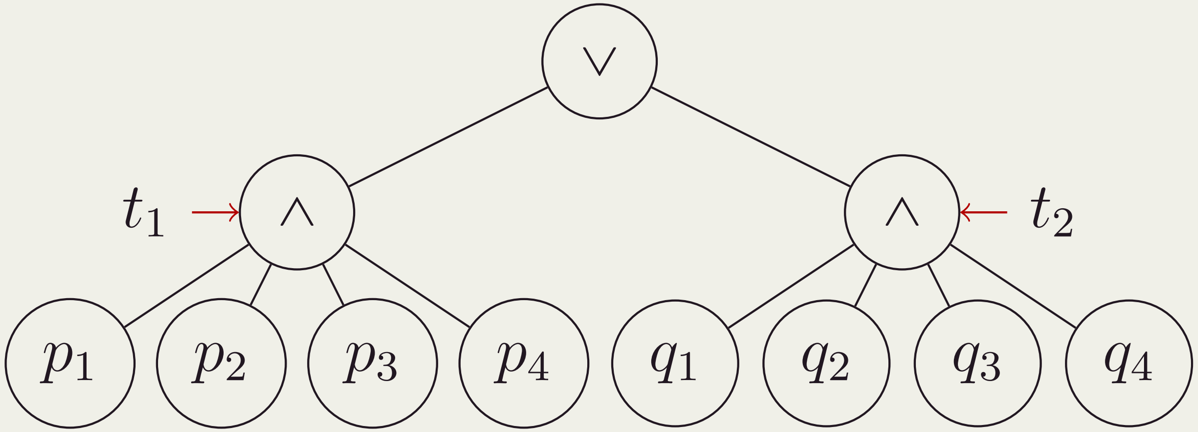
Example



Example

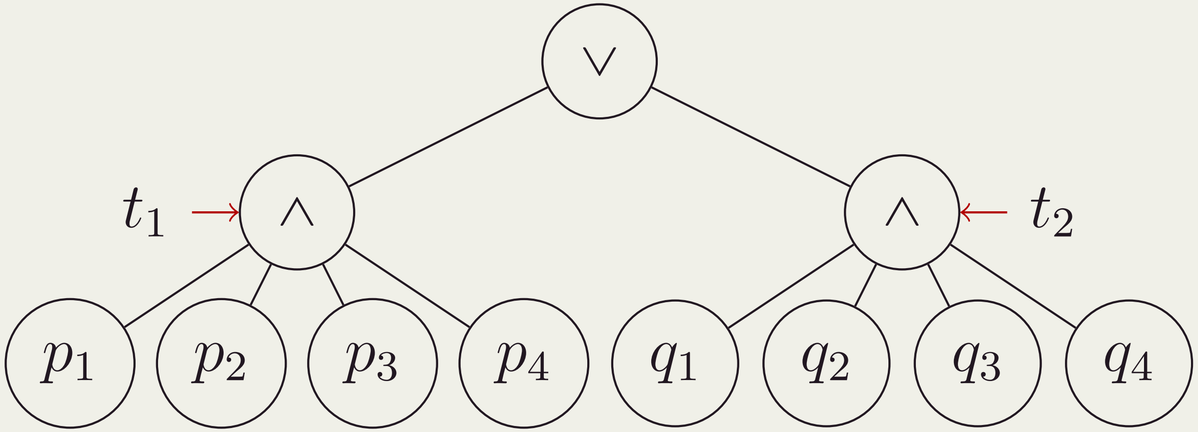


Example



$$(t_1 \vee t_2) \wedge (t_1 \rightarrow p_1 \wedge p_2 \wedge p_3 \wedge p_4) \wedge (t_2 \rightarrow q_1 \wedge q_2 \wedge q_3 \wedge q_4)$$

Example



$$(t_1 \vee t_2) \wedge (t_1 \rightarrow p_1 \wedge p_2 \wedge p_3 \wedge p_4) \wedge (t_2 \rightarrow q_1 \wedge q_2 \wedge q_3 \wedge q_4)$$

$$(t_1 \vee t_2)$$

$$(\neg t_1 \vee p_1) \wedge (\neg t_1 \vee p_2) \wedge (\neg t_1 \vee p_3) \wedge (\neg t_1 \vee p_4)$$

$$(\neg t_2 \vee q_1) \wedge (\neg t_2 \vee q_2) \wedge (\neg t_2 \vee q_3) \wedge (\neg t_2 \vee q_4)$$

More on Renaming

- Renaming ensures that the CNF of a formula has size linear on the size of that formula.
- Renaming helps separating different contexts for reasoning:

$$t \longrightarrow \Diamond_{1} \Diamond_{2} p$$

More on Renaming

- Renaming ensures that the CNF of a formula has size linear on the size of that formula.
- Renaming helps separating different contexts for reasoning:

$$t \longrightarrow \Diamond_1 \Diamond_2 p$$

$$t \longrightarrow \Diamond_1 t_1 \wedge t_1 \longrightarrow \Diamond_2 p$$

More on Renaming

- Renaming ensures that the CNF of a formula has size linear on the size of that formula.
- Renaming helps separating different contexts for reasoning:

$$t \longrightarrow \Diamond_{1} \Diamond_{2} p$$

$$t \longrightarrow \Diamond_{1} t_1 \wedge t_1 \longrightarrow \Diamond_{2} p$$

- In the case of a modal language, we need to make sure that the definition of the new literal is available wherever it is needed:

$$(t \rightarrow \Diamond_{1} t_1) \wedge \Box_{*} (t_1 \rightarrow \Diamond_{2} p)$$

- The use of the universal operator "mimics" the renaming procedure for First-Order Logic, where definitions are universally quantified.

Clauses - Previous Calculus

In [ND, 2006] and [ND, 2007] (inspired by [Mints, 1990])

- Initial clause $\boxed{*}(\mathbf{start} \to \bigvee_{b=1}^{r} l_b)$
- Literal clause $\boxed{*}(\mathbf{true} \to \bigvee_{b=1}^{r} l_b)$
- Positive a-clause $\boxed{*}(l' \to \boxed{a}l)$
- Negative a-clause $\boxed{*}(l' \to \diamondsuit_a l)$

where l, l', $l_b \in \mathcal{L}$. Positive and negative a-clauses are together known as *modal a-clauses*; the index a may be omitted if it is clear from the context.

Modal Layered Clauses

In [NHD, 2015, NDH, 2019] (inspired by [AdNdR, 2000], [AGHdR, 2000]):

- Literal clause $ml : \bigvee_{b=1}^{r} l_b$
- Positive a-clause $ml : l' \rightarrow \boxed{a} l$
- Negative a-clause $ml : l' \rightarrow \Diamond_a l$

where $ml \in \mathbb{N} \cup \{*\}$ and l, l', $l_b \in \mathcal{L}$.

Inference Rules

[LRES]

$$\begin{array}{rlcl} ml: & D & \vee & l \\ ml': & D' & \vee & \neg l \\ \hline \sigma(\{ml, ml'\}): & D & \vee & D' \end{array}$$

[MRES]

$$\begin{array}{rlcl} ml: & l_1 & \rightarrow & \boxed{a} l \\ ml': & l_2 & \rightarrow & \Diamond_a \neg l \\ \hline \sigma(\{ml, ml'\}): & \neg l_1 & \vee & \neg l_2 \end{array}$$

where $\sigma(\{i\}) = i$, $\sigma(\{i, *\}) = i$,
$i \in \{*\} \cup \mathbb{N}$

$0 : p \vee q, 0 : \neg p \vee q$

$* : p \vee q, * : \neg p \vee q$

$* : p \vee q, 1 : \neg p \vee q$

$0 : p \vee q, 1 : \neg p \vee q$

Inference Rules

[LRES]

$$\begin{array}{rrcl} ml: & D & \vee & l \\ ml': & D' & \vee & \neg l \\ \hline \sigma(\{ml, ml'\}): & D & \vee & D' \end{array}$$

[MRES]

$$\begin{array}{rrcl} ml: & l_1 & \rightarrow & \boxed{a} l \\ ml': & l_2 & \rightarrow & \Diamond_a \neg l \\ \hline \sigma(\{ml, ml'\}): & \neg l_1 & \vee & \neg l_2 \end{array}$$

where $\sigma(\{i\}) = i$, $\sigma(\{i, *\}) = i$,
$i \in \{*\} \cup \mathbb{N}$:

$$0 : p \vee q, 0 : \neg p \vee q$$

$$* : p \vee q, * : \neg p \vee q$$

$$* : p \vee q, 1 : \neg p \vee q$$

$$0 : p \vee q, 1 : \neg p \vee q$$

Inference Rules

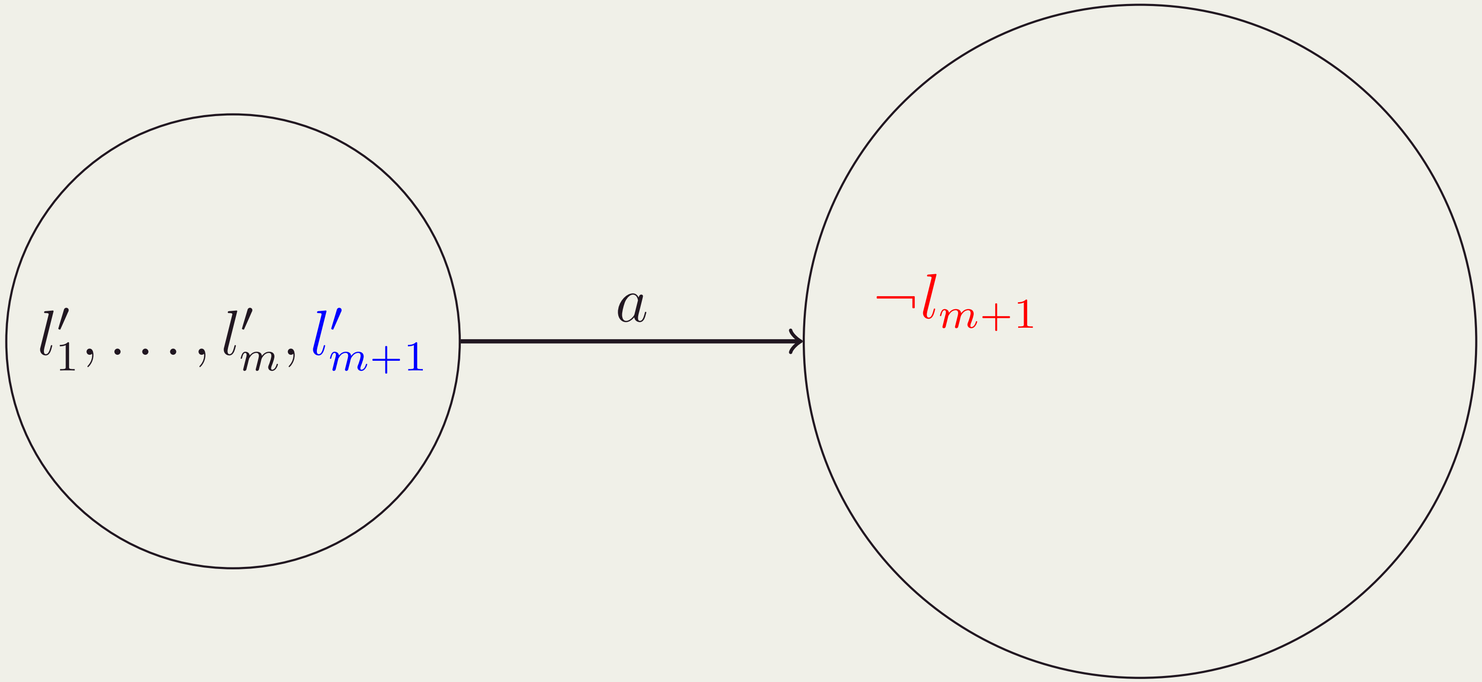
[GEN1]

$$
\begin{array}{rl}
ml_1 : & l'_1 \to [a]\neg l_1 \\
& \vdots \\
ml_m : & l'_m \to [a]\neg l_m \\
ml_{m+1} : & l'_{m+1} \to \langle a \rangle \neg l \\
ml_{m+2} : & l_1 \vee \ldots \vee l_m \vee l_{m+1} \\
\hline
\sigma\left(\{ml_{m+2} - 1\} \cup \bigcup_{i=1}^{m+1}\{ml_i\}\right) : & \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'_{m+1}
\end{array}
$$

$$l'_1, \ldots, l'_m, l'_{m+1}$$

[GEN1]

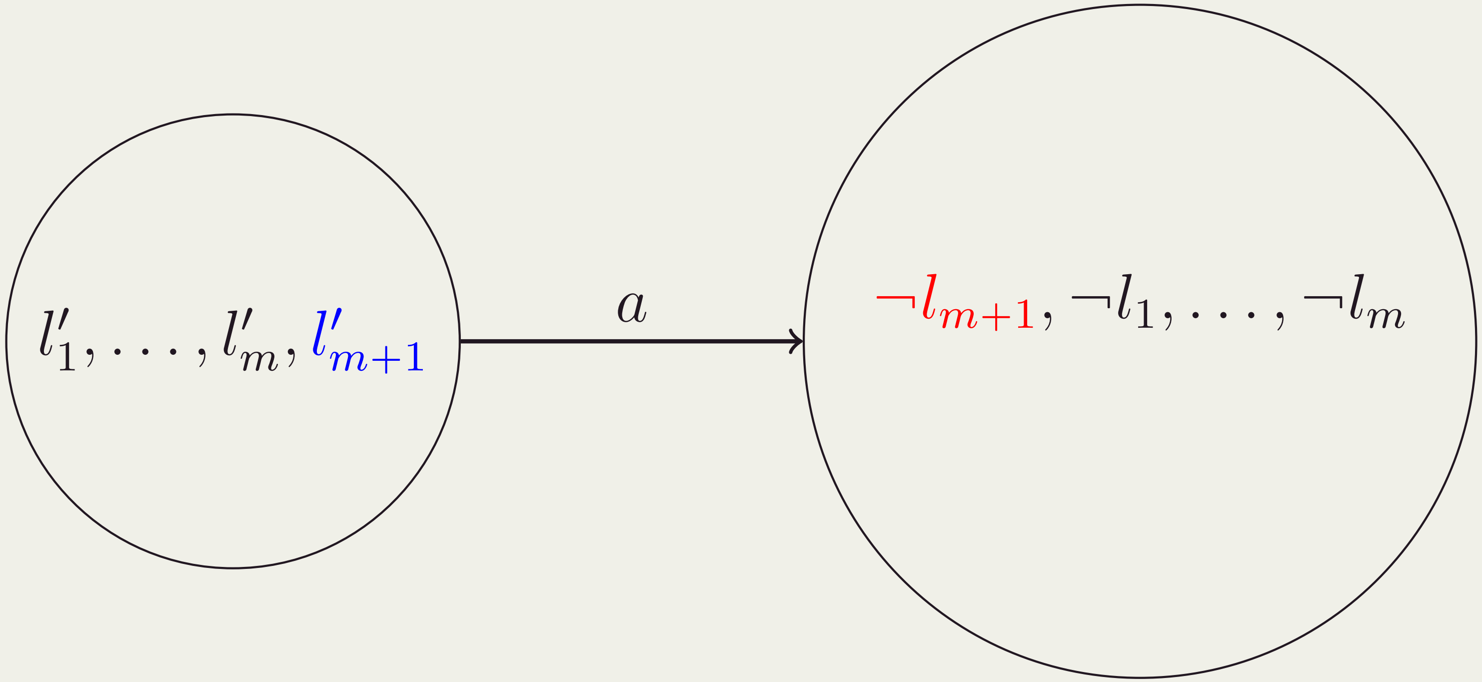
$$
\begin{array}{rl}
ml_1 : & l'_1 \rightarrow \boxed{a}\neg l_1 \\
& \vdots \\
ml_m : & l'_m \rightarrow \boxed{a}\neg l_m \\
ml_{m+1} : & l'_{m+1} \rightarrow \Diamond_a \neg l_{m+1} \\
ml_{m+2} : & l_1 \vee \ldots \vee l_m \vee l_{m+1} \\
\hline
\sigma\left(\{ml_{m+2} - 1\} \cup \bigcup_{i=1}^{m+1}\{ml_i\}\right) : & \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'_{m+1}
\end{array}
$$



Inference Rules

[GEN1]

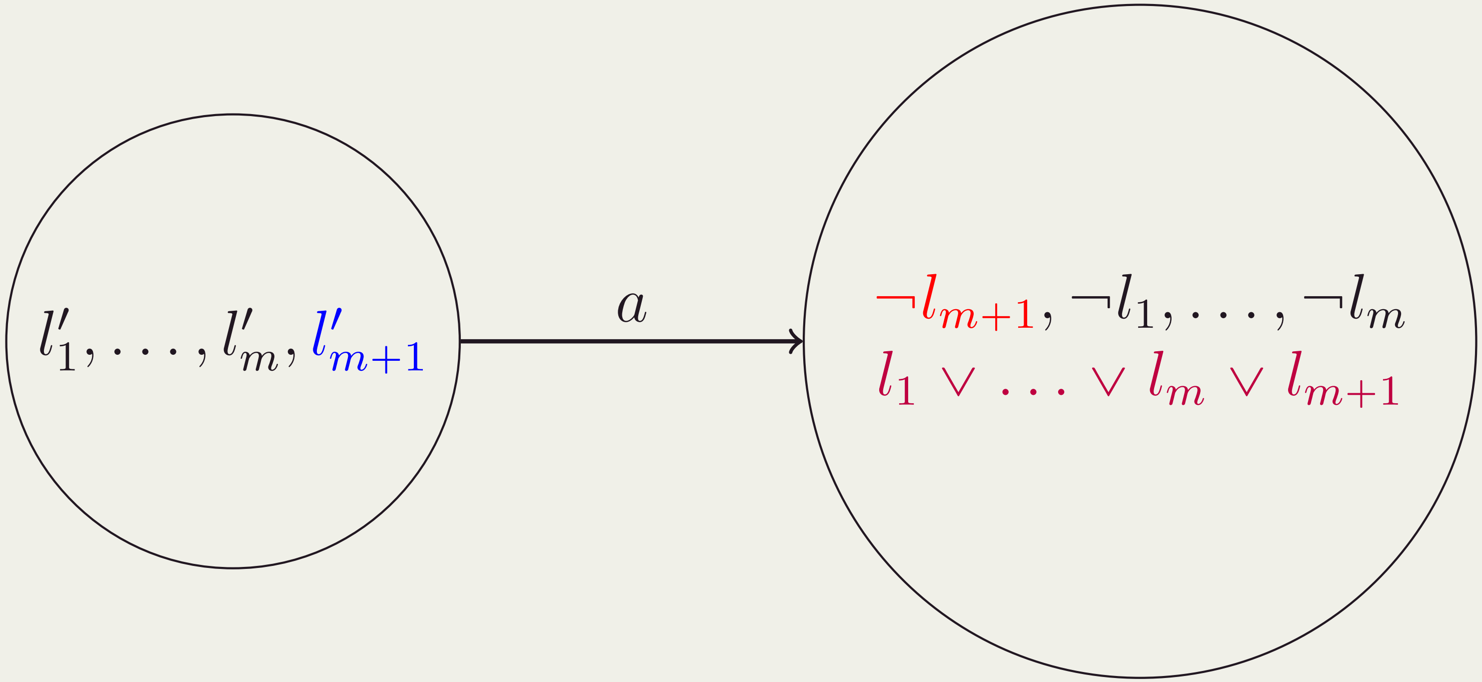
$$
\begin{array}{rl}
ml_1 : & l'_1 \rightarrow \boxed{a}\neg l_1 \\
& \vdots \\
ml_m : & l'_m \rightarrow \boxed{a}\neg l_m \\
ml_{m+1} : & l'_{m+1} \rightarrow \Diamond_a \neg l_{m+1} \\
ml_{m+2} : & l_1 \vee \ldots \vee l_m \vee l_{m+1} \\
\hline
\sigma\left(\{ml_{m+2} - 1\} \cup \bigcup_{i=1}^{m+1}\{ml_i\}\right) : & \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'_{m+1}
\end{array}
$$



Inference Rules

[GEN1]

$$
\begin{array}{rl}
ml_1 : & l'_1 \to \boxed{a}\neg l_1 \\
 & \vdots \\
ml_m : & l'_m \to \boxed{a}\neg l_m \\
ml_{m+1} : & l'_{m+1} \to \langle a \rangle \neg l_{m+1} \\
ml_{m+2} : & l_1 \lor \ldots \lor l_m \lor l_{m+1} \\
\hline
\sigma\left(\{ml_{m+2} - 1\} \cup \bigcup_{i=1}^{m+1}\{ml_i\}\right) : & \neg l'_1 \lor \ldots \lor \neg l'_m \lor \neg l'_{m+1}
\end{array}
$$



Inference Rules

[GEN2]

$$\frac{\begin{array}{rcl} ml_1: & l'_1 & \rightarrow \boxed{a} l_1 \\ ml_2: & l'_2 & \rightarrow \boxed{a} \neg l_1 \\ ml_3: & l'_3 & \rightarrow \Diamond_a l_2 \end{array}}{ml: \quad \neg l'_1 \vee \neg l'_2 \vee \neg l'_3}$$

where $ml = \sigma(\{ml_1, ml_2, ml_3\})$

[GEN3]

$$\frac{\begin{array}{rcl} ml_1: & l'_1 & \rightarrow \boxed{a} \neg l_1 \\ & \vdots & \\ ml_m: & l'_m & \rightarrow \boxed{a} \neg l_m \\ ml_{m+1}: & l' & \rightarrow \Diamond_a l \\ ml_{m+2}: & l_1 \vee \ldots \vee l_m & \end{array}}{ml: \quad \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'}$$

where $ml = \sigma(\{ml_1, \ldots, ml_{m+1}, ml_{m+2} - 1\})$

Examples

$$\Diamond\Diamond p \wedge \Box\neg p$$

$$\begin{aligned}
0 &: \quad t_0 \\
0 &: \quad t_0 \rightarrow \Diamond t_1 \\
1 &: \quad t_1 \rightarrow \Diamond p \\
0 &: \quad t_0 \rightarrow \Box\neg p
\end{aligned}$$

Examples

$$\Diamond\Diamond p \wedge \Box\neg p$$

$$\begin{aligned} 0 &: \quad t_0 \\ 0 &: \quad t_0 \rightarrow \Diamond t_1 \\ 1 &: \quad t_1 \rightarrow \Diamond p \\ 0 &: \quad t_0 \rightarrow \Box\neg p \end{aligned}$$

$$p \wedge \Diamond\neg p$$

$$\begin{aligned} 0 &: \quad t_0 \\ 0 &: \quad \neg t_0 \vee p \\ 0 &: \quad t_0 \rightarrow \Diamond\neg p \end{aligned}$$

Examples

$$\Diamond\Diamond p \wedge \Box\neg p$$

$$\begin{aligned} 0 &: \quad t_0 \\ 0 &: \quad t_0 \rightarrow \Diamond t_1 \\ 1 &: \quad t_1 \rightarrow \Diamond p \\ 0 &: \quad t_0 \rightarrow \Box\neg p \end{aligned}$$

$$p \wedge \Diamond\neg p$$

$$\begin{aligned} 0 &: \quad t_0 \\ 0 &: \quad \neg t_0 \vee p \\ 0 &: \quad t_0 \rightarrow \Diamond\neg p \end{aligned}$$

$$p \wedge \Diamond\neg p$$

$$\begin{aligned} * &: \quad t_0 \\ * &: \quad \neg t_0 \vee p \\ * &: \quad t_0 \rightarrow \Diamond\neg p \end{aligned}$$

Implementation

Overview

K_SP [NHD, 2016, NHD, 2020]:

- Set-of-support (given-clause, as in Otter), but there is one set of support for each modal level;

Overview

K$_S$P [NHD, 2016, NHD, 2020]:

- Set-of-support (given-clause, as in Otter), but there is one set of support for each modal level;
- Refinements: negative, ordered, negative+ordered, ordered with selection, positive resolution;

Overview

K_SP [NHD, 2016, NHD, 2020]:

- Set-of-support (given-clause, as in Otter), but there is one set of support for each modal level;
- Refinements: negative, ordered, negative+ordered, ordered with selection, positive resolution;
- Pre-processing: simplification, pure literal elimination, modal level pure literal elimination, unit propagation, populating automatically the usable, different techniques for renaming, prenex/antiprenex, cnf;

Overview

K$_S$P [NHD, 2016, NHD, 2020]:

- Set-of-support (given-clause, as in Otter), but there is one set of support for each modal level;
- Refinements: negative, ordered, negative+ordered, ordered with selection, positive resolution;
- Pre-processing: simplification, pure literal elimination, modal level pure literal elimination, unit propagation, populating automatically the usable, different techniques for renaming, prenex/antiprenex, cnf;
- Redundancy elimination: (lazy) forward/backward subsumption, pure literal elimination, modal level pure literal elimination . . .

Overview

K$_S$P [NHD, 2016, NHD, 2020]:

- Set-of-support (given-clause, as in Otter), but there is one set of support for each modal level;
- Refinements: negative, ordered, negative+ordered, ordered with selection, positive resolution;
- Pre-processing: simplification, pure literal elimination, modal level pure literal elimination, unit propagation, populating automatically the usable, different techniques for renaming, prenex/antiprenex, cnf;
- Redundancy elimination: (lazy) forward/backward subsumption, pure literal elimination, modal level pure literal elimination . . .
- Clause selection: shortest, newest, oldest, greatest literal, smallest literal.

Overview

K$_S$P [NHD, 2016, NHD, 2020]:

- Set-of-support (given-clause, as in Otter), but there is one set of support for each modal level;
- Refinements: negative, ordered, negative+ordered, ordered with selection, positive resolution;
- Pre-processing: simplification, pure literal elimination, modal level pure literal elimination, unit propagation, populating automatically the usable, different techniques for renaming, prenex/antiprenex, cnf;
- Redundancy elimination: (lazy) forward/backward subsumption, pure literal elimination, modal level pure literal elimination . . .
- Clause selection: shortest, newest, oldest, greatest literal, smallest literal.

The full pack is in my webpage: nalon.org.

K_SP- LWB - k_t4p - Modal Layering

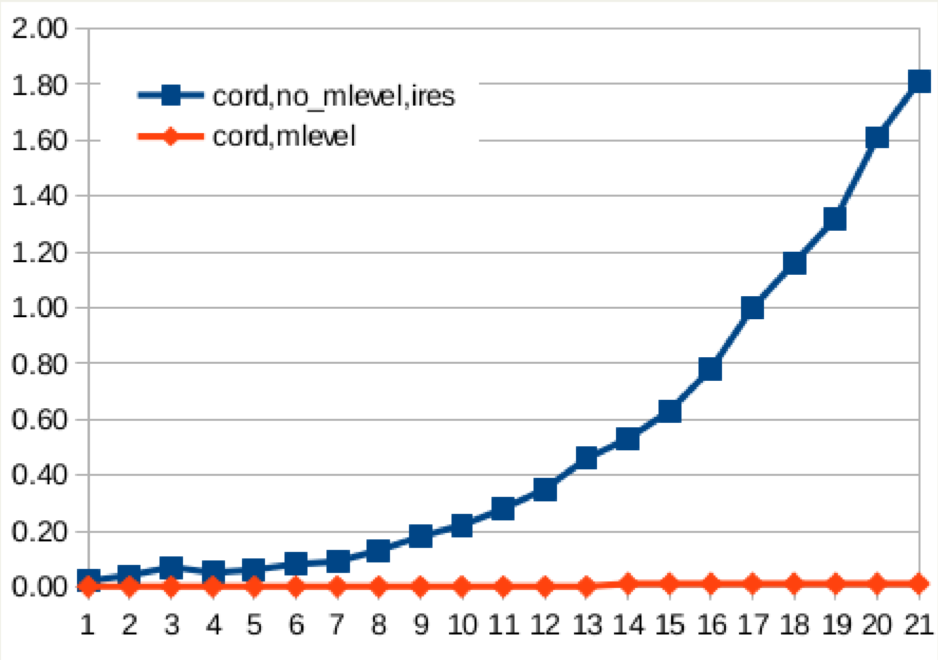


Figura 1: Unsatisfiable Formulae

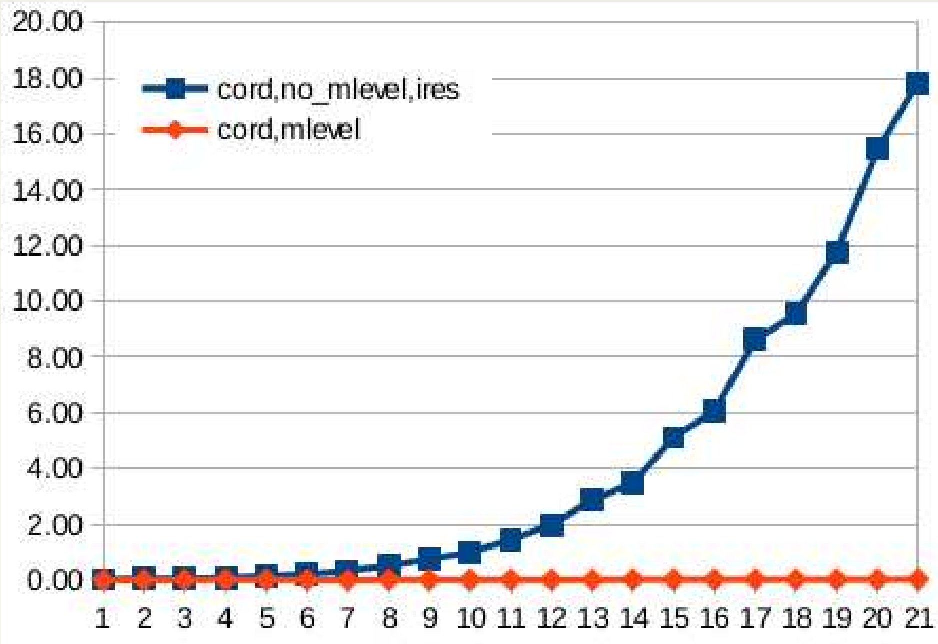
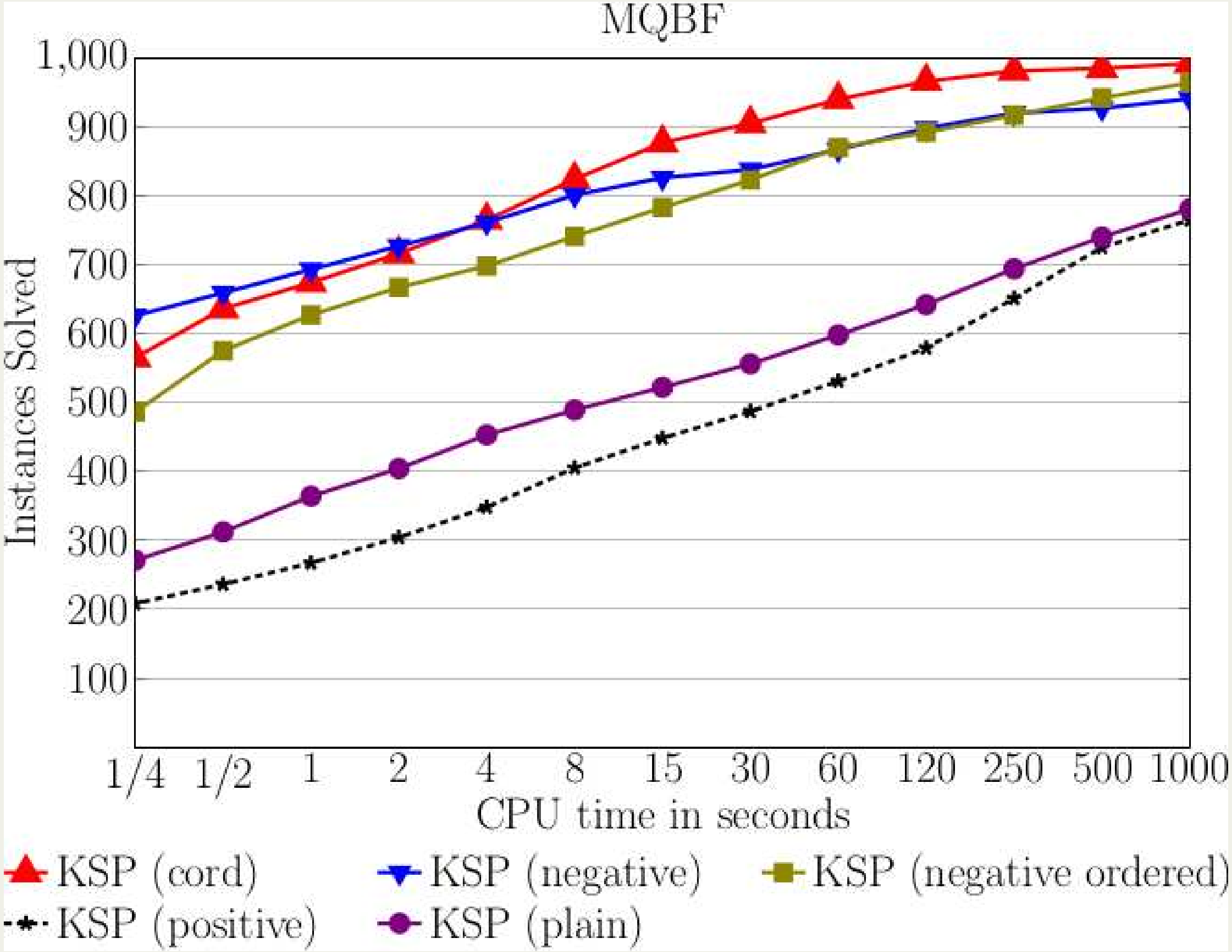
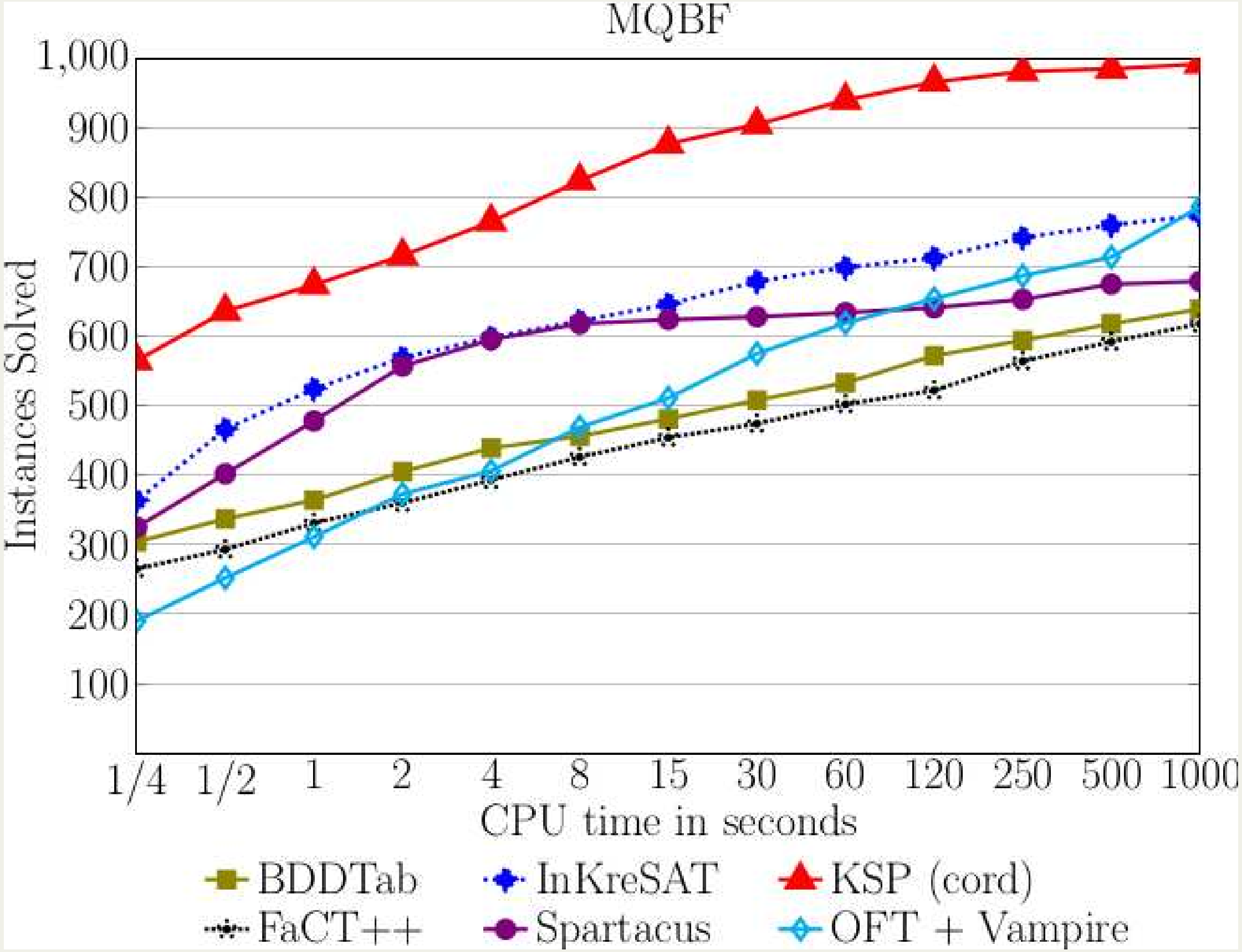


Figura 2: Satisfiable Formulae

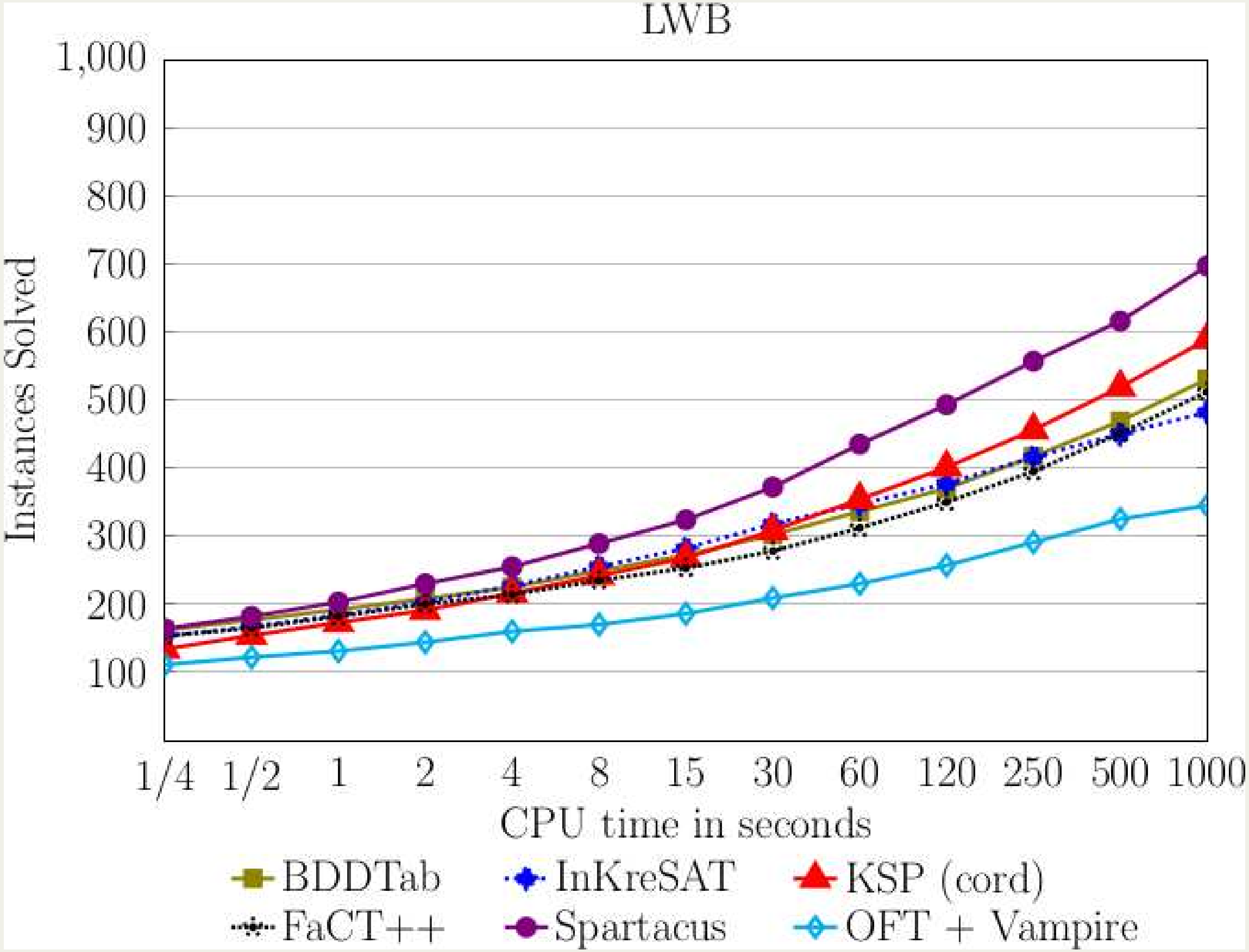
K_SP- MQBF - Different Refinements



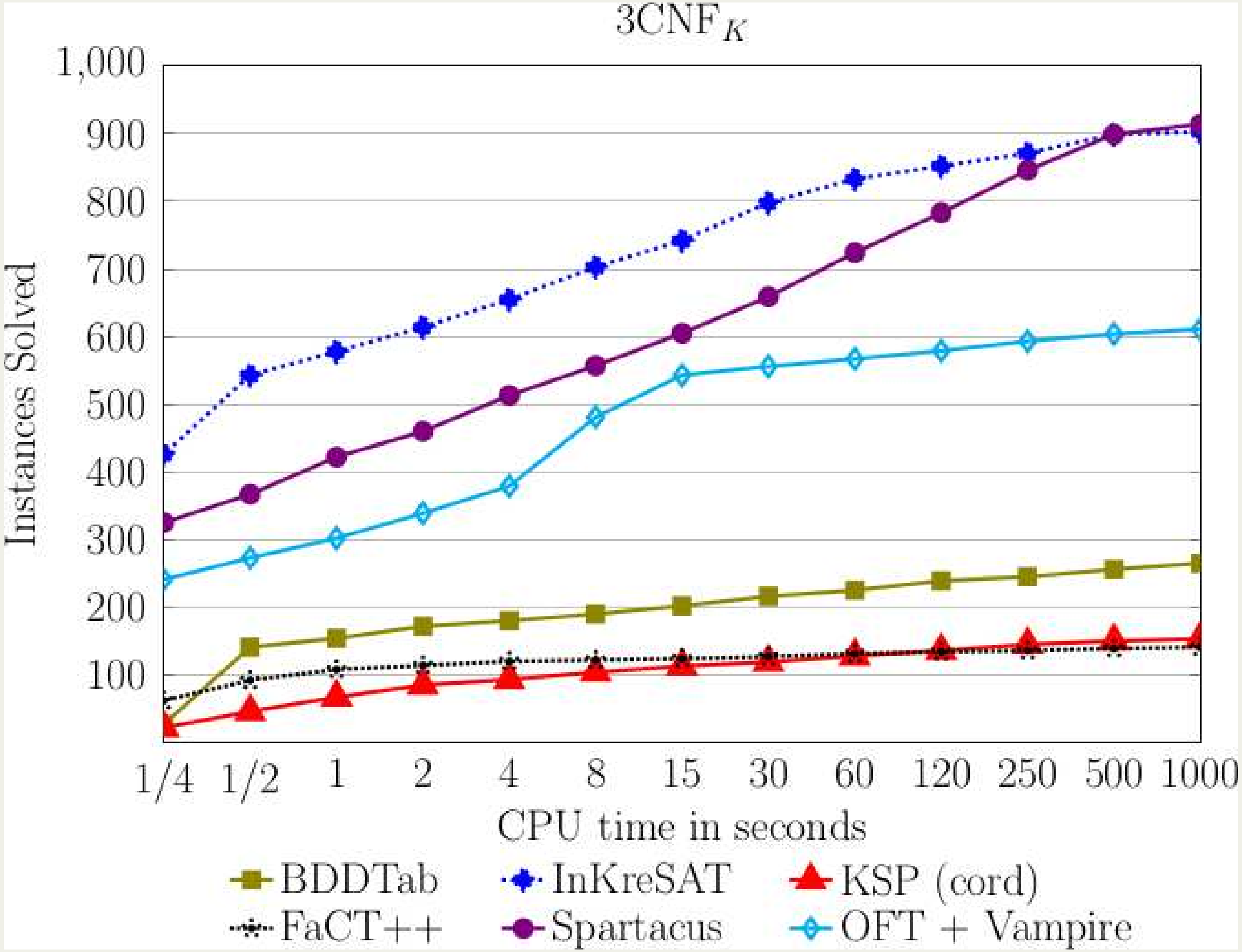
All Provers - MQBF



All Provers - LWB

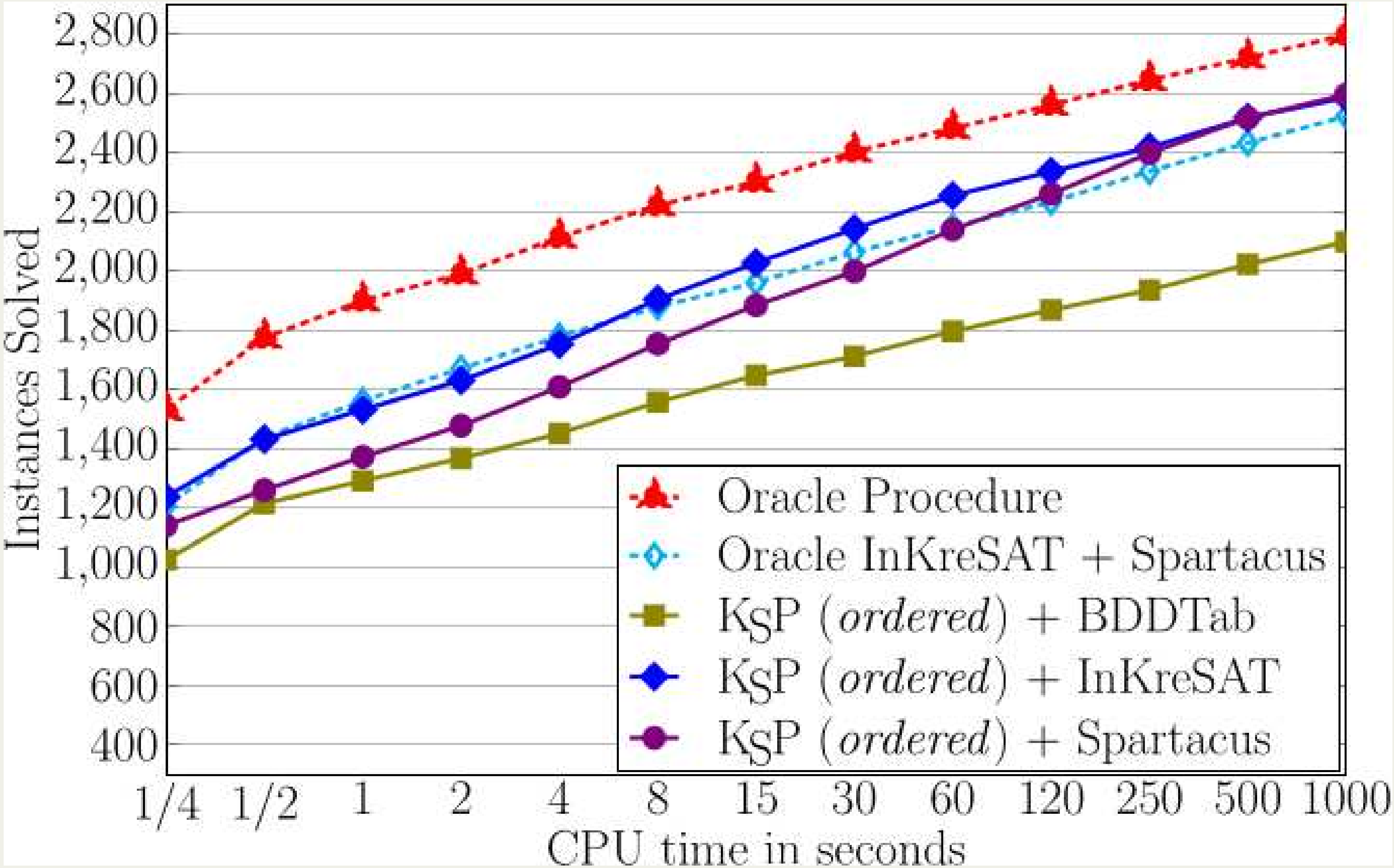


All Provers - 3CNF



Oracle/Portfolio

BDDTab	FaCT++	InKreSAT	KSP	Spartacus	OFT + Vampire	Unsolved
674	111	912	849	748	57	227



Some Notes

- The calculus is sound, complete, and terminating (TABLEAUX 2015, ToCL 2020).
- The calculus for K_n was implemented and tested (IJCAR 2016, JAR 2020).
- Negative and ordered resolution, together with layering, are also complete (ToCL 2020).
- Ongoing and future work:
  - KSP is not any clever (yet).
  - Renaming can be improved????
  - Saturation takes a lot of time: combined proof methods might help here.

References

[Halpern and Moses, 1992] Halpern, J. Y. and Moses, Y. (1992). A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54(3):319–379.

[Ladner, 1977] Ladner, R. E. (1977). The computational complexity of provability in systems of modal propositional logic. *SIAM J. Comput.*, 6(3):467–480.

[Mints, 1990] Mints, G. (1990). Gentzen-type systems and resolution rules, part I: Propositional logic. *Lecture Notes in Computer Science*, 417:198–231.

[Spaan, 1993] Spaan, E. (1993). *Complexity of Modal Logics*. PhD thesis, University of Amsterdam.

References

[NHD, 2015] Nalon, C., Hustadt, U., and Dixon, C. (2015a). A modal-layered resolution calculus for K. In [Nivelle, 2015], pages 185–200.

[Nivelle, 2015] Nivelle, H. D., editor (2015). *Automated Reasoning with Analytic Tableaux and Related Methods - 24th International Conference, TABLEAUX 2015, Wrocław, Poland, September 21-24, 2015. Proceedings*, volume 9323 of *Lecture Notes in Computer Science*. Springer.

[NHD, 2016] Nalon, C., Hustadt, U., and Dixon, C. (2016). K_SP: A resolution-based prover for multimodal K. In Olivetti, N. and Tiwari, A., editors, *Automated Reasoning: 8th International Joint Conference, IJCAR 2016, Coimbra, Portugal, June 27 – July 2, 2016, Proceedings*, pages 406–415, Cham. Springer International Publishing.

[NDH, 2019] Nalon, C., Dixon, C., Hustadt, U.: Modal resolution: Proofs, layers, and refinements. ACM Trans. Comput. Log. **20**(4), 23:1–23:38 (2019)

[NHD, 2020] Nalon, C., Hustadt, U., Dixon, C.: K_SP: Architecture, refinements, strategies and experiments. J. Autom. Reason. **64**(3), 461–484 (2020)

References

[ND, 2006] Nalon, C. and Dixon, C. (2006). Anti-prenexing and prenexing for modal logics. In *Proceedings of the 10th ECAI*, Liverpool, UK.

[ND, 2007] Nalon, C. and Dixon, C. (2007). Clausal resolution for normal modal logics. *J. Algorithms*, 62:117–134.

[NMD, 2014] C. Nalon, J. Marcos, and C. Dixon. Clausal resolution for modal logics of confluence. In S. Demri, D. Kapur, and C. Weidenbach, editors, *Automated Reasoning. Proceedings of the 7th International Joint Conference on Automated Reasoning (IJCAR)*, volume 8562 of *Lecture Notes in Computer Science*, pages 322–336. Springer, 2014.

References

[Tseitin,1968] G. Tseitin. On the complexity of derivations in the propositional logics. In A. O. Slisenko, editor, *Studies in Constructive Mathematics and Mathematical Logic, Part II*, pages 115–125. 1968.

[PG, 1986] D. A. Plaisted and S. A. Greenbaum. A Structure-Preserving Clause Form Translation. *Journal of Logic and Computation*, 2:293–304, 1986.

[Haken, 1985] A. Haken. The Intractability of Resolution. *Theoretical Computer Science*, 39:297–308, 1985.

References

[AdNdR, 2000] C. Areces, H. de Nivelle, and M. de Rijke. Prefixed Resolution: A Resolution Method for Modal and Description Logics. In H. Ganzinger, editor, *Proc. CADE-16*, volume 1632 of *LNAI*, pages 187–201, Berlin, July 7–10 1999. Springer.

[AGHdR, 2000] C. Areces, R. Gennari, J. Heguiabehere, and M. D. Rijke. Tree-based heuristics in modal theorem proving. In *Proc. of ECAI 2000*, pages 199–203. IOS Press, 2000.

[AH, 2002] C. Areces and J. Heguiabehere. HyLoRes: A hybrid logic prover, Sept. 18 2002.

[AG, 2011] C. Areces and D. Gorín. Resolution with order and selection for hybrid logics. *Journal of Automated Reasoning*, 46(1):1–42, 2011.

	BDDTab		FaCT++		InKreSAT		K_SP (cord)		Spartacus		OFT + Vampire	
branch_n	22	*22*	12	*12*	15	*15*	18	*18*	12	*12*	**50**	*70*
branch_p	22	*22*	12	*12*	22	*22*	23	*24*	14	*14*	**50**	*70*
d4_n	20	*440*	6	*40*	34		**48**	*1560*	28	*760*	14	*200*
d4_p	26	*640*	24	*600*	18	*360*	**54**	*1800*	32	*920*	21	*960*
dum_n	39	*2400*	42	*2640*	23	*1120*	**49**	*3200*	44	*2800*	17	*640*
dum_p	42	*2640*	38	*2320*	28	*1520*	**50**	*3280*	46	*2960*	18	*720*
grz_n	35	*2600*	27	*1800*	50	*4500*	5	*50*	**52**	*5500*	24	*1500*
grz_p	35	*2600*	27	*1800*	51	*5000*	29	*2000*	**52**	*5500*	27	*1800*
lin_n	46	*4000*	43	*3400*	33	*2500*	1	*10*	**50**	*4800*	40	*3100*
lin_p	14	*500*	28	*10000*	**56**	*500000*	23	*5000*	55	*400000*	28	*10000*
path_n	37	*290*	48	*400*	7	*14*	**54**	*1000*	47	*400*	41	*330*
path_p	35	*270*	48	*400*	5	*12*	**54**	*1000*	47	*400*	41	*330*
ph_n	10	*10*	8	*16*	24	*90*	3	*6*	**21**	*75*	15	*45*
ph_p	**11**	*11*	9	*8*	10	*10*	5	*5*	9	*9*	10	*10*
poly_n	39	*600*	34	*500*	30		36	*540*	**44**	*720*	20	*220*
poly_p	38	*580*	34	*500*	28	*400*	36	*540*	**44**	*700*	20	*220*
t4p_n	40	*3500*	24	*1500*	17	*800*	39	*3000*	**45**	*6000*	11	*200*
t4p_p	48	*7500*	49	*8000*	28		49	*8000*	**53**	*12000*	14	*500*