Combined Proof Methods for Multimodal Logic

Daniella Angelos

March 20, 2018

Cláudia Nalon — University of Brasilia

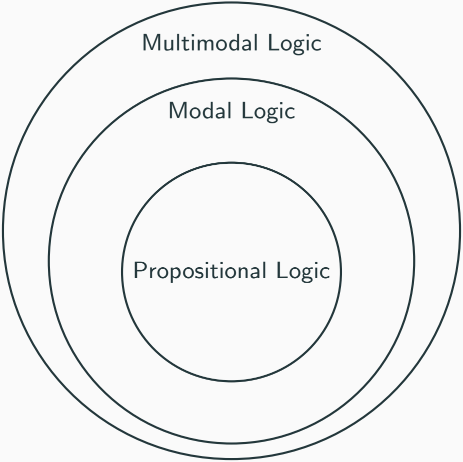
Introduction

Logic

in Computer Science:

Develop languages to model situations

Languages



Multimodal Logic (K_n)

- Propositional Symbols ($\mathcal{P} = \{p, q, r \ldots\}$)
- Operators ($\neg, \vee, \boxed{a}$) with $a \in \mathcal{A} = \{1, \ldots, n\}$

Multimodal Logic (K_n)

Well Formed Formulae (WFF)

- Propositional Symbols $\mathcal{P}$
- If φ and ψ are each a WFF then also are:
  1. $\neg\varphi$
  2. $\varphi \vee \psi$
  3. $\boxed{a}\varphi$

Semantics

Let W be a set of possible worlds and R_a a binary relation over these worlds for some $a \in \mathcal{A}$:

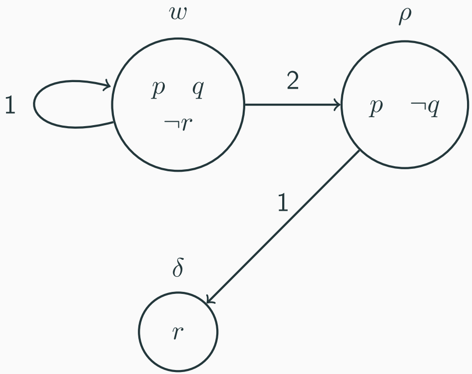
> Possibility: $\Diamond_a \varphi$ is true at $w \in W$ iff φ is true at some $w' \in W$ such that wR_aw'

Semantics

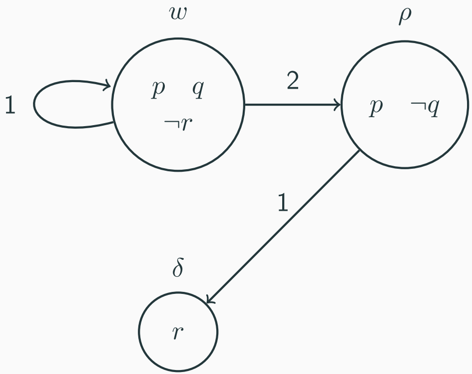
Let W be a set of possible worlds and R_a a binary relation over these worlds for some $a \in \mathcal{A}$:

> Possibility: $\langle a \rangle \varphi$ is true at $w \in W$ iff φ is true at some $w' \in W$ such that wR_aw'
>
> Necessity: $\boxed{a}\varphi$ is true at $w \in W$ iff φ is true at all $w' \in W$ such that wR_aw'

Example



Example



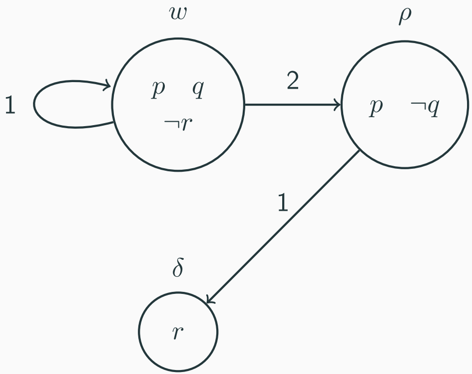
Formulae true in w:

$\boxed{1}\, p$

$\boxed{1}\, q \wedge \boxed{2}\, \neg q$

$\langle 2 \rangle \langle 1 \rangle r$

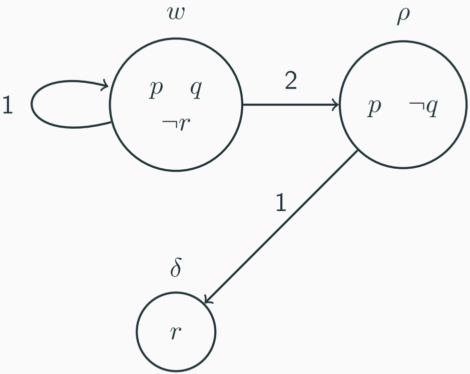
Example



Formulae true in w:

- $\boxed{1}\, p$: modal level of $p = 1$
- $\boxed{1}\, q \wedge \boxed{2}\, \neg q$
- $\Diamond_2 \Diamond_1 r$

Example



Formulae true in w:

- $\boxed{1}\,p$: modal level of $p = 1$
- $\boxed{1}\,q \wedge \boxed{2}\,\neg q$
- $\langle 2 \rangle \langle 1 \rangle r$: modal level of $r = 2$

Model - Kripke Structure

$$\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$$

Model - Kripke Structure

$$\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$$

- Set of possible worlds W

Model - Kripke Structure

$$\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$$

- Set of possible worlds W
- Root of model $w_0 \in W$

Model - Kripke Structure

$$\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$$

- Set of possible worlds W
- Root of model $w_0 \in W$
- A finite number of binary relations R_a over the worlds in W, $R_a \subseteq W \times W$, with $a \in \mathcal{A}$

Model - Kripke Structure

$$\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$$

- Set of possible worlds W
- Root of model $w_0 \in W$
- A finite number of binary relations R_a over the worlds in W, $R_a \subseteq W \times W$, with $a \in \mathcal{A}$
- Valuation function to propositional symbols $\pi : W \times \mathcal{P} \rightarrow \{true, false\}$

Satisfiability Relation

Let φ be a WFF and $\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$ a model in K_n:

Satisfiability Relation

Let φ be a WFF and $\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$ a model in K_n:

$\langle \mathfrak{M}, w \rangle \models \varphi$

Satisfiability Relation

Let φ be a WFF and $\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$ a model in K_n:

$\langle \mathfrak{M}, w \rangle \models \varphi$

- $\langle \mathfrak{M}, w \rangle \models p$ iff $\pi(w, p) = true$

Satisfiability Relation

Let φ be a WFF and $\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$ a model in K_n:

$\langle \mathfrak{M}, w \rangle \models \varphi$

- $\langle \mathfrak{M}, w \rangle \models p$ iff $\pi(w, p) = true$
- $\langle \mathfrak{M}, w \rangle \models \neg\psi$ iff $\langle \mathfrak{M}, w \rangle \not\models \psi$

Satisfiability Relation

Let φ be a WFF and $\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$ a model in K_n:

$\langle \mathfrak{M}, w \rangle \models \varphi$

- $\langle \mathfrak{M}, w \rangle \models p$ iff $\pi(w, p) = true$
- $\langle \mathfrak{M}, w \rangle \models \neg\psi$ iff $\langle \mathfrak{M}, w \rangle \not\models \psi$
- $\langle \mathfrak{M}, w \rangle \models \psi \vee \delta$ iff $\langle \mathfrak{M}, w \rangle \models \psi$ or $\langle \mathfrak{M}, w \rangle \models \delta$

Satisfiability Relation

Let φ be a WFF and $\mathfrak{M} = (W, w_0, R_1, \ldots, R_n, \pi)$ a model in K_n:

$\langle \mathfrak{M}, w \rangle \models \varphi$

- $\langle \mathfrak{M}, w \rangle \models p$ iff $\pi(w, p) = true$
- $\langle \mathfrak{M}, w \rangle \models \neg\psi$ iff $\langle \mathfrak{M}, w \rangle \not\models \psi$
- $\langle \mathfrak{M}, w \rangle \models \psi \vee \delta$ iff $\langle \mathfrak{M}, w \rangle \models \psi$ or $\langle \mathfrak{M}, w \rangle \models \delta$
- $\langle \mathfrak{M}, w \rangle \models \boxed{a}\,\psi$ iff for all $w' \in W$ such that $wR_a w'$, with $a \in \mathcal{A}$, we have that $\langle \mathfrak{M}, w' \rangle \models \psi$

Satisfiability

A formula $\varphi \in \mathsf{WFF}$ is said to be *locally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w_0 \rangle \models \varphi$.

Satisfiability

A formula $\varphi \in \mathrm{WFF}$ is said to be *locally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w_0 \rangle \models \varphi$.

$\mathfrak{M} \models_L \varphi$

Satisfiability

A formula $\varphi \in \text{WFF}$ is said to be *locally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w_0 \rangle \models \varphi$.

$\mathfrak{M} \models_L \varphi$

A formula $\varphi \in \text{WFF}$ is said to be *globally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w \rangle \models \varphi$ for all w.

Satisfiability

A formula $\varphi \in \mathsf{WFF}$ is said to be *locally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w_0 \rangle \models \varphi$.

$\mathfrak{M} \models_L \varphi$

A formula $\varphi \in \mathsf{WFF}$ is said to be *globally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w \rangle \models \varphi$ for all w.

$\mathfrak{M} \models_G \varphi$

Satisfiability

A formula $\varphi \in$ WFF is said to be *locally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w_0 \rangle \models \varphi$.

$\mathfrak{M} \models_L \varphi$

A formula $\varphi \in$ WFF is said to be *globally satisfiable* in K_n if exists a model $\mathfrak{M}$ such that $\langle \mathfrak{M}, w \rangle \models \varphi$ for all w.

$\mathfrak{M} \models_G \varphi$

A set $\mathcal{F}$ of WFF is said to be (locally of globally) satisfiable, if the conjunction of each $\varphi \in \mathcal{F}$ is satisfiable.

Problem

Given any set Φ of well formed formulae, to determine if Φ is (locally or globally) satisfiable in K_n.

Problem

Given any set Φ of well formed formulae, to determine if Φ is (locally or globally) satisfiable in K_n.

- Local satisfiability: PSPACE-Complete (Halpern and Moses – 1992),

Problem

Given any set Φ of well formed formulae, to determine if Φ is (locally or globally) satisfiable in K_n.

- Local satisfiability: PSPACE-Complete (Halpern and Moses – 1992),
- Global satisfiability: EXPTIME-Complete (E. Spaan – 1993).

Layered Resolution

Clauses in $\mathsf{SNF}_{\mathsf{K}_n}$

$$\bigwedge_i ml : C_i$$

Clauses in $\mathsf{SNF}_{\mathsf{K}_n}$

$$\bigwedge_i ml : C_i$$

- literal clause:

$$ml : \bigvee_{b=1}^{r} l_b$$

Clauses in $\mathsf{SNF}_{\mathsf{K}_n}$

$$\bigwedge_i ml : C_i$$

- literal clause:

$$ml : \bigvee_{b=1}^{r} l_b$$

- a-positive clause:

$$ml : l' \Rightarrow \boxed{a}\, l$$

Clauses in $\mathsf{SNF}_{\mathsf{K}_n}$

$$\bigwedge_i ml : C_i$$

- literal clause:

$$ml : \bigvee_{b=1}^{r} l_b$$

- a-positive clause:

$$ml : l' \Rightarrow \boxed{a}\, l$$

- a-negative clause:

$$ml : l' \Rightarrow \Diamond_a l$$

Clauses in $\mathsf{SNF}_{\mathsf{K}_n}$

$$\bigwedge_i ml : C_i$$

- literal clause:

$$ml : \bigvee_{b=1}^{r} l_b$$

- a-positive clause:

$$ml : l' \Rightarrow \boxed{a}\, l$$

- a-negative clause:

$$ml : l' \Rightarrow \langle a \rangle\, l$$

where l is a propositional symbol or its negation

Calculus

$$[\text{LRES}] \quad \frac{\begin{array}{ll} ml: & D \vee l \\ ml: & D' \vee \neg l \end{array}}{\begin{array}{ll} ml: & D \vee D' \end{array}}$$

Calculus

[GEN1]

$$\begin{array}{rl}
ml: & l'_1 \Rightarrow \boxed{a}\neg l_1 \\
\vdots & \\
ml: & l'_m \Rightarrow \boxed{a}\neg l_m \\
ml: & l' \Rightarrow \Diamond_a \neg l \\
ml+1: & l_1 \vee \ldots \vee l_m \vee l \\
\hline
ml: & \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'
\end{array}$$

[GEN3]

$$\begin{array}{rl}
ml: & l'_1 \Rightarrow \boxed{a}\neg l_1 \\
\vdots & \\
ml: & l'_m \Rightarrow \boxed{a}\neg l_m \\
ml: & l' \Rightarrow \Diamond_a l \\
ml+1: & l_1 \vee \ldots \vee l_m \\
\hline
ml: & \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l'
\end{array}$$

K$_S$P

- A theorem prover that implements this modal resolution-based calculus
- It also implements several refinements and simplification techniques
- K$_S$P performs pretty well!
- But not so well when there is a large number of variables in one particular level

- A theorem prover that implements this modal resolution-based calculus
- It also implements several refinements and simplification techniques
- $\text{K}_\text{S}\text{P}$ performs pretty well!
- But not so well when there is a large number of variables in one particular level

Problem

As resolution relies on saturation, the performance of $\text{K}_\text{S}\text{P}$ for such entries deteriorates.

Hypotheses

We take advantage of the great theoretical and practical efforts that have been directed in improving the efficiency of CDCL SAT solvers, to reduce the time KςP spends during saturation.

CDCL SAT Solvers

SAT Problem

Given a formula in classical propositional logic, does this formula have a satisfying assignment?

SAT Solvers

SAT Solvers

- Generic combinatorial tool and search platform

SAT Solvers

- Generic combinatorial tool and search platform
- DPLL Procedure

SAT Solvers

- Generic combinatorial tool and search platform
- DPLL Procedure
- Watched literals, restart strategies, deletion mechanisms, branching heuristics, learning mechanisms etc

Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s)$$

Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s)$$

$$p = false$$

Example

$$\varphi = (p \lor t \lor \neg q) \land (p \lor \neg r) \land (q \lor r \lor s)$$

$$p = false \longrightarrow r = false$$

Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s)$$

$t = false$

$p = false \longrightarrow r = false$

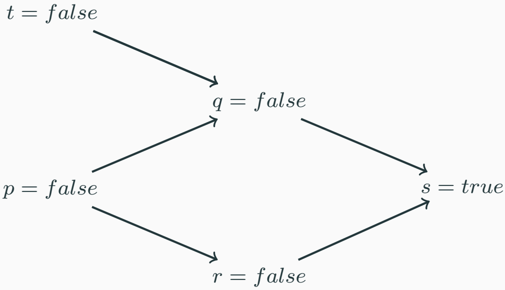
Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s)$$

$$\begin{tikzcd}
t = false \arrow[dr] & \\
& q = false \\
p = false \arrow[ur] \arrow[dr] & \\
& r = false
\end{tikzcd}$$

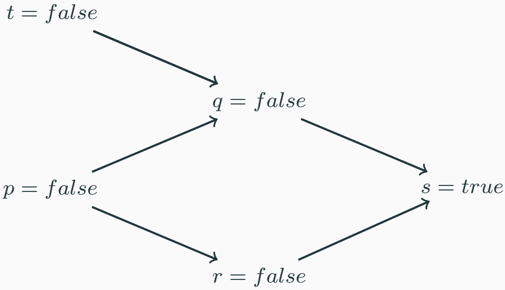
Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s)$$



Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s) \wedge (\neg s \vee \neg u) \wedge (y \vee x) \wedge (y \vee u \vee \neg x)$$



Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s) \wedge (\neg s \vee \neg u) \wedge (y \vee x) \wedge (y \vee u \vee \neg x)$$

$$\begin{tikzcd}
t = false \arrow[dr] & & & \\
& q = false \arrow[dr] & & u = false \\
p = false \arrow[ur] \arrow[dr] & & s = true \arrow[ur] & \\
& r = false \arrow[ur] & &
\end{tikzcd}$$

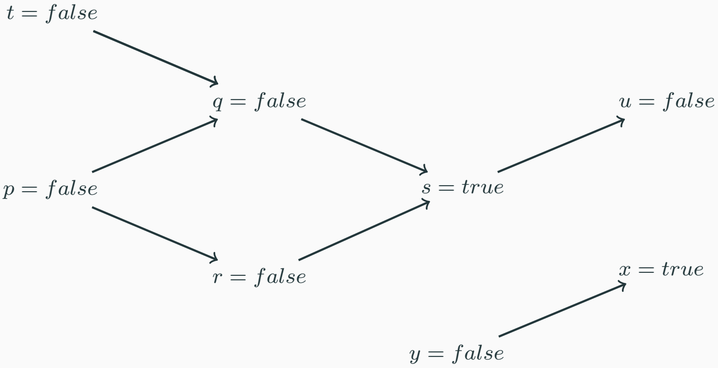
Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s) \wedge (\neg s \vee \neg u) \wedge (y \vee x) \wedge (y \vee u \vee \neg x)$$

$$\begin{tikzcd}
t = false \arrow[dr] & & & \\
& q = false \arrow[dr] & & u = false \\
p = false \arrow[ur] \arrow[dr] & & s = true \arrow[ur] & \\
& r = false \arrow[ur] & & \\
& & y = false &
\end{tikzcd}$$

Example

$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s) \wedge (\neg s \vee \neg u) \wedge (y \vee x) \wedge (y \vee u \vee \neg x)$$



Example

$$\varphi =(p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s) \wedge (\neg s \vee \neg u) \wedge (y \vee x) \wedge (y \vee u \vee \neg x)$$

$$\begin{tikzcd}
t = false \arrow[dr] & & & & \\
& q = false \arrow[dr] & & u = false \arrow[dr] & \\
p = false \arrow[ur] \arrow[dr] & & s = true \arrow[ur] & & \varepsilon \\
& r = false \arrow[ur] & & x = true \arrow[ur] & \\
& & y = false \arrow[ur] & &
\end{tikzcd}$$

Implication Graph

Implication Graph

- Vertices are assigned variables

Implication Graph

- Vertices are assigned variables
- Edges are antecedent clauses

Conflict analysis

Analyses the conflict and learns a new clause

Conflict analysis

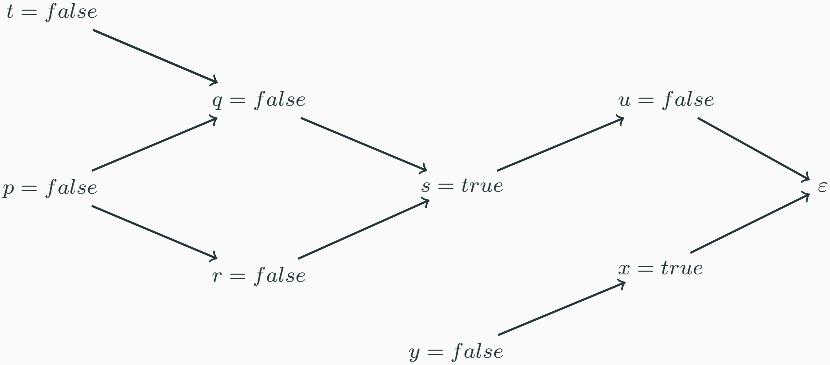
$$\varphi = (p \vee t \vee \neg q) \wedge (p \vee \neg r) \wedge (q \vee r \vee s) \wedge (\neg s \vee \neg u) \wedge (y \vee x) \wedge (y \vee u \vee \neg x)$$

$$\begin{tikzcd}
t = false \arrow[dr] & & & & \\
& q = false \arrow[dr] & & u = false \arrow[dr] & \\
p = false \arrow[ur] \arrow[dr] & & s = true \arrow[ur] & & \varepsilon \\
& r = false \arrow[ur] & & x = true \arrow[ur] & \\
& & y = false \arrow[ur] & &
\end{tikzcd}$$

Conflict analysis

$$\varphi = (p \lor t \lor \neg q) \land (p \lor \neg r) \land (q \lor r \lor s) \land (\neg s \lor \neg u) \land (y \lor x) \land (y \lor u \lor \neg x)$$

new clause: $(p \lor t \lor y)$



Combination

K_SP + CDCL

During the main loop of K_SP, we will feed the SAT solver with the propositional clauses at a particular modal level.

K$_S$P + CDCL

During the main loop of K$_S$P, we will feed the SAT solver with the propositional clauses at a particular modal level.

- The SAT solver finds a model for the set of clauses

K_SP + CDCL

During the main loop of K_SP, we will feed the SAT solver with the propositional clauses at a particular modal level.

- The SAT solver finds a model for the set of clauses
- The SAT solver returns one or more new clauses

Example

$$1 : l_1 \Rightarrow \Box \neg p$$
$$1 : l_2 \Rightarrow \Box \neg t$$
$$1 : l_3 \Rightarrow \Diamond \neg y$$
$$2 : \{p, t, \neg q\}$$
$$2 : \{p, \neg r\}$$
$$2 : \{q, r, s\}$$
$$2 : \{\neg s, \neg u\}$$
$$2 : \{y, x\}$$
$$2 : \{y, u, \neg x\}$$

Example

$$1 : l_1 \Rightarrow \Box \neg p$$
$$1 : l_2 \Rightarrow \Box \neg t$$
$$1 : l_3 \Rightarrow \Diamond \neg y$$
$$2 : \{p, t, \neg q\}$$
$$2 : \{p, \neg r\}$$
$$2 : \{q, r, s\}$$
$$2 : \{\neg s, \neg u\}$$
$$2 : \{y, x\}$$
$$2 : \{y, u, \neg x\}$$

[GEN1]

$$\begin{array}{rl} ml : & {l'}_1 \Rightarrow \boxed{a} \neg l_1 \\ \vdots & \\ ml : & {l'}_m \Rightarrow \boxed{a} \neg l_m \\ ml : & l' \Rightarrow \Diamond_a \neg l \\ ml + 1 : & l_1 \vee \ldots \vee l_m \vee l \\ \hline ml : & \neg {l'}_1 \vee \ldots \vee \neg {l'}_m \vee \neg l' \end{array}$$

Example

$1 : l_1 \Rightarrow \Box \neg p$

$1 : l_2 \Rightarrow \Box \neg t$

$1 : l_3 \Rightarrow \Diamond \neg y$

$2 : \{p, t, \neg q\}$

$2 : \{p, \neg r\}$

$2 : \{q, r, s\}$

$2 : \{\neg s, \neg u\}$

$2 : \{y, x\}$

$2 : \{y, u, \neg x\}$

$2 : \{p, t, y\}$

[GEN1]

$$\frac{\begin{array}{rl} ml: & l'_1 \Rightarrow \boxed{a}\neg l_1 \\ \vdots & \\ ml: & l'_m \Rightarrow \boxed{a}\neg l_m \\ ml: & l' \Rightarrow \Diamond_a \neg l \\ ml+1: & l_1 \vee \ldots \vee l_m \vee l \end{array}}{\begin{array}{rl} ml: & \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l' \end{array}}$$

Example

$1 : l_1 \Rightarrow \Box \neg p$
$1 : l_2 \Rightarrow \Box \neg t$
$1 : l_3 \Rightarrow \Diamond \neg y$
$2 : \{p, t, \neg q\}$
$2 : \{p, \neg r\}$
$2 : \{q, r, s\}$
$2 : \{\neg s, \neg u\}$
$2 : \{y, x\}$
$2 : \{y, u, \neg x\}$
$2 : \{p, t, y\}$
$1 : \{\neg l_1, \neg l_2, \neg l_3\}$

[GEN1]

$$\begin{array}{rl} ml : & l'_1 \Rightarrow \boxed{a} \neg l_1 \\ \vdots & \\ ml : & l'_m \Rightarrow \boxed{a} \neg l_m \\ ml : & l' \Rightarrow \Diamond_a \neg l \\ ml + 1 : & l_1 \vee \ldots \vee l_m \vee l \\ \hline ml : & \neg l'_1 \vee \ldots \vee \neg l'_m \vee \neg l' \end{array}$$

Conclusion

Conclusion

We believe that calling the SAT solver from the main loop of KSP will significantly improve its performance.

Future Work

Future Work

1. Choose a SAT solver and evaluate this choice

Future Work

1. Choose a SAT solver and evaluate this choice
2. Automated integration with KSP

Future Work

1. Choose a SAT solver and evaluate this choice
2. Automated integration with KSP
3. Define metrics for tests and select benchmarks

Future Work

1. Choose a SAT solver and evaluate this choice
2. Automated integration with KSP
3. Define metrics for tests and select benchmarks
4. Exhaustively testing

Future Work

1. Choose a SAT solver and evaluate this choice
2. Automated integration with KSP
3. Define metrics for tests and select benchmarks
4. Exhaustively testing
5. Collect and analyse data

Future Work

1. Choose a SAT solver and evaluate this choice
2. Automated integration with KSP
3. Define metrics for tests and select benchmarks
4. Exhaustively testing
5. Collect and analyse data
6. Write the dissertation

Future Work

1. Choose a SAT solver and evaluate this choice
2. Automated integration with KSP
3. Define metrics for tests and select benchmarks
4. Exhaustively testing
5. Collect and analyse data
6. Write the dissertation

Activities	Months									
	March	April	May	June	July	August	September	October	November	December
1	x	x								
2	x	x	x	x						
3				x	x					
4					x	x	x	x		
5							x	x	x	
6	x	x	x	x	x	x	x	x	x	x

Thank you!

Combined Proof Methods for Multimodal Logic

Daniella Angelos

March 20, 2018

Cláudia Nalon — University of Brasilia