Kripke's World

An introduction to modal logics via tableau systemss

O. Gasquet, A. Herzig, B. Saïd, F. Schwarzentruber

Institut de Recherche en Informatique de Toulouse - IRIT
Université de Toulouse
http://www.irit.fr/Lotrec

UNILOG 2010

Background: logic and reasoning

- Classical propositional logic (CPL)
  - satisfiability problem decidable: NP-complete
  - reasoning:
    - Hilbert-style axiomatics, natural deduction
    - Gentzen sequent systems, tableaux
    - *resolution*
    - *heuristic search: many SAT solvers*
- Classical predicate logic
  - satisfiability problem semi-decidable
  - reasoning:
    - ...
    - *resolution [OTTER, SPASS, etc.]*
- Higher-order logic
  - undecidable
  - reasoning:
    - *Proof assistants [Isabelle, Coq, etc.]*

Background and motivation

- Modal logics
  - variant: description logics ($\Longrightarrow$ semantic web)
  - infinitely many logics
  - 'surprisingly often decidable'
    - NP $<$ PSPACE $<$ EXPTIME $<$ NEXPTIME $<$ EXPSPACE
  - reasoning:
    - Hilbert-style axiomatics, natural deduction
    - Gentzen sequent systems
    - resolution [Fariñas 83]
    - translation to FOL and resolution [Fariñas and Herzig 88, Ohlbach 88; MSPASS]
    - methods based on SAT solvers for CPL [K-SAT, etc.]
    - *Tableaux*

Idea: step-by-step introduction to modal logics via tableaux

From Tarski's World to Kripke's World

- Tarski's World: introduction to predicate logic
  - examples = scenarios from geometry
  - book + program
- Kripke's World: introduction to modal logics
  - examples = modal logics
  - reasoning = try to construct models = tableaux
  - program: LoTREC, `http://www.irit.fr/Lotrec`
  - book to come

Outline

Part 1: Theory

1 Modal logics

2 Reasoning problems

Part 2: Practice

3 LoTREC

4 Implementing logics

Part 1: Theory

1 Modal logics
- Possible worlds models
- Classes of models
- Language
- Semantics

2 Reasoning problems
- Validity and satisfiability in a class of models
- Outline of the tableaux method

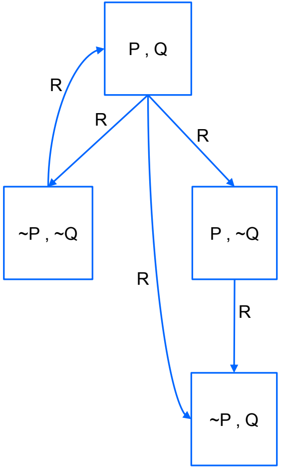
Outline

1 Modal logics
- Possible worlds models
- Classes of models
- Language
- Semantics

2 Reasoning problems
- Validity and satisfiability in a class of models
- Outline of the tableaux method

What is a Kripke model?

- Possible worlds
  = node
  = states
- Valuation
  = labeling function
  = interpretation
- Accessibility relation
  = labeled edges
  = transitions
- Model
  = labeled graph
  = transition system



Kripke Model

Given: a set $\mathcal{P}$ (propositional variables) and a set $\mathcal{I}$ (indexes):

- $M = (W, R, V)$
  - W: nonempty set — set of possible worlds
  - R: $\mathcal{I} \longrightarrow 2^{W \times W}$ — accessibility relation
  - V: $W \longrightarrow 2^{\mathcal{P}}$ — valuation function
- Pointed model (M, w)
  where $w \in W$ is the actual world

Readings of R

- Alethic:
  wRu iff u is possible given the actual world w
- Temporal:
  wRu iff u is in the future of w
- Epistemic:
  $wR_I u$ iff u is possible for agent I at actual world w
- Deontic:
  wRu iff u is an ideal counterpart of the actual world w
- Dynamic:
  $wR_I u$ iff u is a possible result of the execution of the program / action I in w

Readings of $R \Longrightarrow$ Properties of R

Readings of R

- Alethic:
  wRu iff u is possible given the actual world w
- Temporal:
  wRu iff u is in the future of w
- Epistemic:
  $wR_I u$ iff u is possible for agent I at actual world w
- Deontic:
  wRu iff u is an ideal counterpart of the actual world w
- Dynamic:
  $wR_I u$ iff u is a possible result of the execution of the program / action I in w

Readings of $R \Longrightarrow$ Properties of R

Defining a model in LoTREC

How to build a graph with two nodes:

- open a new logic (menu 'Logic')
- add a new rule ('Rules' tab):
  - no conditions
  - in the action part:

```
createNewNode w
createNewNode u
link w u R
add w P
```

- edit the default strategy ('Strategies' tab):
  - call the new rule (double click)

Outline

1. Modal logics
   - Possible worlds models
   - Classes of models
   - Language
   - Semantics
2. Reasoning problems
   - Validity and satisfiability in a class of models
   - Outline of the tableaux method

Classes of models

- A class of models can be defined by
  - constraints on the accessibility relation
  - constraints on the valuation
- Applications?
- Mathematical properties?

Constraints on a single relation R

- Singleton models:
  $\{M : card(W) = 1\}$
- Serial
  'there is always a future'
  for all w exists u s.th. wRu
- Reflexive
  'knowledge implies truth'
- Transitive
  'future of future is future'
  'I know what I know'
- Symmetric
- Euclidian
  'I know what I don't know'
- Confluent *(Church-Rosser)*
- Equivalence
- Universal
- ...

Constraints involving several relations

- R_I included in R_J
- $R_I = R_J \cup R_K$
- $R_J = (R_I)^{-1}$
- $R_J = (R_I)^*$ *(reflexive and transitive closure)*
- $R_I \circ R_J = R_J \circ R_I$ *(permutation)*
- Confluent
- ...

Constraints on the valuation V

- names for worlds ('nominals'):
  if $N \in V(w)$ and $N \in V(u)$ then $w = u$
  $\Longrightarrow$ hybrid logic
- R is hereditary (atomic propositions persist)
  if $P \in V(w)$ and wRu then $P \in V(u)$
  $\Longrightarrow$ intuitionistic logic

Closing under constraints in LoTREC

- Closing under reflexivity:
  condition: `isNewNode w`
  action: `link w w R`
- Observe:
  capital first letter $\Longrightarrow$ constant
  small first letter $\Longrightarrow$ variable
- Exercise: make `R` hereditary

Outline

1 Modal logics
- Possible worlds models
- Classes of models
- Language
- Semantics

2 Reasoning problems
- Validity and satisfiability in a class of models
- Outline of the tableaux method

Boolean formulas

- atomic formulas = elements of $\mathcal{P}$ (propositional variables)
- complex formulas: built using the Boolean connectors

$$
\begin{array}{lcl}
\neg A & = & \text{“not } A\text{”} \\
A \wedge B & = & \text{“} A \text{ and } B\text{”} \\
A \vee B & = & \text{“} A \text{ or } B\text{”} \\
A \rightarrow B & = & \text{“if } A \text{ then } B\text{”} \\
A \leftrightarrow B & = & \text{“} A \text{ if and only if } B\text{”} \\
A \oplus B & = & \text{“either } A \text{ or } B\text{”} \\
\oplus(A, B, C) & = & \text{“either } A\text{, or } B\text{, or } C\text{”} \\
 & \ldots &
\end{array}
$$

Modal formulas

- Temporal logic

$$\begin{array}{lcl} \mathrm{X}A & = & \text{“}A \text{ will be true at the } \textit{next} \text{ time point”} \\ \mathrm{F}A & = & \text{“}A \text{ will be true at } \textit{some} \text{ time point in the future”} \\ & = & \text{“}A \text{ will eventually be true”} \\ \mathrm{G}A & = & \text{“}A \text{ will be true at } \textit{every} \text{ time point in the future”} \\ & = & \text{“}A \text{ will be true henceforth”} \\ A\mathrm{U}B & = & \text{“}A \text{ until } B\text{”} \\ A\mathrm{S}B & = & \text{“}A \text{ since } B\text{”} \end{array}$$

- Dynamic logic

$$\begin{array}{lcl} \mathrm{After}_I A & = & \text{“}A \text{ will be true after } \textit{every possible} \text{ execution of program } I\text{”} \\ \mathrm{Feasible}_I A & = & \text{“}A \text{ will be true after } \textit{some} \text{ execution of program } I\text{”} \end{array}$$

(programs may be nondeterministic)

Modal formulas (ctd.)

- Epistemic and doxastic logic
  $\mathrm{Bel}_I A$ = "agent I believes that A"
  $\mathrm{K}_I A$ = "agent I knows that A"
  $\hat{\mathrm{Bel}}_I A$ = "it is (doxastically) possible for agent I that A"
  $\hat{\mathrm{K}}_I A$ = "it is (epistemically) possible for agent I that A"
- Deontic logic
  $\mathrm{O}_I B$ = "A is obligatory for I"
  $\mathrm{P}_I B$ = "A is permitted for I"
- Intuitionistic logic
  $A \Rightarrow B$ = "*A implies B*" (like $\rightarrow$, but no excluded middle)
- Conditional logic
  $A \Rightarrow B$ = "*A implies B*" ($\Rightarrow$ 'stronger' than $\rightarrow$)
- ...

"Un pour tous, tous pour un" [A. Dumas]

- An abstraction: necessity and possibility
  $\Diamond A \quad = \quad \mathrm{M}A \quad = \quad$ "A is possible"
  $\Box A \quad = \quad \mathrm{L}A \quad = \quad$ "A is necessary"
- Multimodal version:
  $\Diamond_I A \quad = \quad \langle I \rangle A \quad = \quad$ "A is possible w.r.t. I"
  $\Box_I A \quad = \quad [I]A \quad = \quad \ldots$
  where $I \in \mathcal{I}$ (set of parameters)
- Common feature: Not truth-functional
  - no f s.th. $truthvalue(\Diamond A) = f(truthvalue(A))$

Duality

- Intuitively:

$$\begin{array}{lll} \hat{\mathrm{K}}_I A & \leftrightarrow & \neg \mathrm{K}_I \neg A \\ \mathrm{P}_I A & \leftrightarrow & \neg \mathrm{O}_I \neg A \\ \mathrm{F} A & \leftrightarrow & \neg \mathrm{G} \neg A \\ \mathrm{After}_I A & \leftrightarrow & \neg \mathrm{Feasible}_I \neg A \\ \ldots & & \end{array}$$

- Abstracting:

$$\begin{array}{lll} \Diamond A & \leftrightarrow & \neg \Box \neg A \\ \Box A & \leftrightarrow & \neg \Diamond \neg A \end{array}$$

- Options:
  - take both $\Diamond$ and $\Box$ as primitive
  - take $\Diamond$ as primitive, and set $\Box A \stackrel{\text{def}}{=} \neg \Diamond \neg A$
  - take $\Box$ as primitive, and set $\Diamond A \stackrel{\text{def}}{=} \neg \Box \neg A$

How define a language?

- Examples
  - $\mathit{CardRed} \wedge \mathrm{K}_{\mathit{Ann}} \mathit{CardRed} \wedge \mathrm{K}_{\mathit{Ann}} \neg \mathrm{K}_{\mathit{Bob}} \mathit{CardRed}$
  - $\mathit{DoorClosed} \wedge [\mathit{Open}]\mathit{DoorOpen}$
  - $P \wedge \neg Q \wedge \Box Q \wedge \Diamond(P \wedge \Box \neg Q)$
- Language = set of formulas
- Language is defined by BNF:

$$A ::= P \mid \neg A \mid A \wedge A \mid A \vee A \mid \Diamond A \mid \Box A \mid \langle I \rangle A \mid [I]A \mid \mathrm{K}_I A \mid \ldots$$

where P ranges over $\mathcal{P}$ and I ranges over $\mathcal{I}$

How define a language in LoTREC?

- Formulas in LoTREC: prenex form
  $\Longrightarrow$ General schema: $op(A_1, \ldots, A_n)$

$$\begin{array}{rcl} \neg A & = & not(A) \\ A \wedge B & = & and(A, B) \\ A \vee B & = & or(A, B) \\ & \ldots & \end{array}$$

$$\begin{array}{rcl} \mathrm{Bel}_I A & = & Bel(I, A) \\ \mathrm{K}_I A & = & Knows(I, A) \\ \hat{\mathrm{K}}_I A & = & Poss(I, A) \\ & \ldots & \\ A \mathrm{U} B & = & Until(A, B) \\ & \ldots & \\ A \Rightarrow B & = & ifThen(A, B) \\ & \ldots & \end{array}$$

- A LoTREC formula is
  - a propositional variable $P \in \mathcal{P}$, or
  - an expression of the form $op(A_1, \ldots, A_n)$ where op is the name of a logical connector and the A_i are formulas or in $\mathcal{I}$

  $\Longrightarrow$ General schema: $op(Arg_1, \ldots, Arg_n)$, where $Arg_i \in \mathcal{P} \cup \mathcal{I}$

Outline

1 Modal logics
- Possible worlds models
- Classes of models
- Language
- Semantics

2 Reasoning problems
- Validity and satisfiability in a class of models
- Outline of the tableaux method

Truth conditions

- Atoms
  - $M, w \Vdash P$ iff $P \in V(w)$
- Classical connectors
  - $M, w \Vdash A \wedge B$ iff $M, w \Vdash A$ and $M, w \Vdash B$
  - $M, w \Vdash A \vee B$ iff ...
  - ...
- Modal operators
  - $M, w \Vdash \Diamond A$ iff there exists u s.th. wRu and $M, u \Vdash A$
  - $M, w \Vdash \Box A$ iff for all u, if wRu then $M, u \Vdash A$

Truth conditions

- Multi-modal operators
  - $M, w \Vdash \langle I \rangle A$ iff there exists u s.th. $w R_I u$ and $M, u \Vdash A$
  - ...
- Relation algebra operators
  - $M, w \Vdash \Diamond^{-1} A$ iff there exists u s.th. $w R^{-1} u$ and $M, u \Vdash A$
  - $M, w \Vdash \langle I \cup J \rangle A$ iff there exists u s.th. $w (R_I \cup R_J) u$ and $M, u \Vdash A$
  - $M, w \Vdash \langle I^* \rangle A$ iff there exists u s.th. $w (R_I)^* u$ and $M, u \Vdash A$

Truth conditions

- Temporal operators (linear time)
  - $M, w \Vdash \mathrm{X}A$ iff there exists u s.th. wRu and $M, u \Vdash A$
  - $M, w \Vdash \mathrm{F}A$ iff there exists n,u s.th. $wR^n u$ and $M, u \Vdash A$

$$\underset{w}{A} \longrightarrow A \longrightarrow \ldots \quad \longrightarrow A \longrightarrow \underset{u}{B}$$

  - $M, w \Vdash A\mathrm{U}B$ iff there exists u s.th. wR^*u and $M, u \Vdash B$ and $M, v \Vdash A$ for all v s.th. $(wR^*v$ and $vR^+u)$
  - ...

Model checking

$\boxed{\text{Given } M, w, \text{ and } A, \text{ do we have } M, w \Vdash A \text{ ?}}$

- Model checking problem
  - can be solved in polynomial time for most modal logics
- Model checking in LoTREC
  - requires more LoTREC primitives $\Longrightarrow$ later

Part 1: Theory

1 Modal logics
- Possible worlds models
- Classes of models
- Language
- Semantics

2 Reasoning problems
- Validity and satisfiability in a class of models
- Outline of the tableaux method

Outline

1. Modal logics
   - Possible worlds models
   - Classes of models
   - Language
   - Semantics

2. Reasoning problems
   - Validity and satisfiability in a class of models
   - Outline of the tableaux method

Validity and satisfiability in the set of all models

K = the set of all possible worlds models (**K**ripke)

- A is valid in K iff *for all* M in K and *all* w in M: $M, w \Vdash A$

Example

- $\Box(P \vee \neg P)$
- $\Box P \wedge \Box Q \rightarrow \Box(P \wedge Q)$

- A is satisfiable in K iff *for some* M in K and *some* w in M: $M, w \Vdash A$

Example

- P
- $P \wedge \neg\Box P$
- $P \wedge \Box\neg P$
- $\Box P \wedge \neg\Box\Box P$

Validity and satisfiability in the set of all models

K = the set of all possible worlds models (**K**ripke)

- A is valid in K iff *for all* M in K and *all* w in M: $M, w \Vdash A$

Example

- $\Box(P \vee \neg P)$
- $\Box P \wedge \Box Q \to \Box(P \wedge Q)$

- A is satisfiable in K iff *for some* M in K and *some* w in M: $M, w \Vdash A$

Example

- P
- $P \wedge \neg\Box P$
- $P \wedge \Box\neg P$
- $\Box P \wedge \neg\Box\Box P$

Validity and satisfiability in some class of models

$\mathcal{C}$ = some subset of K

- A is valid in $\mathcal{C}$ iff *for all* M in $\mathcal{C}$ and *all* w in M: $M, w \Vdash A$

Example

- $\Box P \to P$ invalid in K $\quad \Box P, \neg P \longrightarrow P$
- $\Box P \to P$ valid in the class of reflexive models $\quad \neg P, \cancel{\Box P}$
- $\Diamond\Diamond P \to \Diamond P$ valid in transitive models

- A is satisfiable in $\mathcal{C}$ iff *for some* M in $\mathcal{C}$ and *some* w in M: $M, w \Vdash A$

Example

- $P \wedge \Box\neg P$ is satisfiable in K
- $P \wedge \Box\neg P$ is unsatisfiable in the class of reflexive models

$\boxed{A \text{ is valid in } \mathcal{C} \text{ iff } \neg A \text{ is unsatisfiable in } \mathcal{C}}$

Validity and satisfiability in some class of models

$\mathcal{C}$ = some subset of K

- A is valid in $\mathcal{C}$ iff *for all* M in $\mathcal{C}$ and *all* w in M: $M, w \Vdash A$

Example

- $\Box P \to P$ invalid in K $\quad \Box P, \neg P \longrightarrow P$
- $\Box P \to P$ valid in the class of reflexive models $\quad \neg P, \cancel{\Box P}$
- $\Diamond\Diamond P \to \Diamond P$ valid in transitive models

- A is satisfiable in $\mathcal{C}$ iff *for some* M in $\mathcal{C}$ and *some* w in M: $M, w \Vdash A$

Example

- $P \wedge \Box\neg P$ is satisfiable in K
- $P \wedge \Box\neg P$ is unsatisfiable in the class of reflexive models

A is valid in $\mathcal{C}$ iff $\neg A$ is unsatisfiable in $\mathcal{C}$

Validity and satisfiability in some class of models

$\mathcal{C}$ = some subset of K

- A is valid in $\mathcal{C}$ iff *for all* M in $\mathcal{C}$ and *all* w in M: $M, w \Vdash A$

Example

- $\Box P \to P$ invalid in K $\Box P, \neg P \longrightarrow P$
- $\Box P \to P$ valid in the class of reflexive models $\neg P, \cancel{\Box P}$
- $\Diamond\Diamond P \to \Diamond P$ valid in transitive models

- A is satisfiable in $\mathcal{C}$ iff *for some* M in $\mathcal{C}$ and *some* w in M: $M, w \Vdash A$

Example

- $P \land \Box\neg P$ is satisfiable in K
- $P \land \Box\neg P$ is unsatisfiable in the class of reflexive models

$\boxed{A \text{ is valid in } \mathcal{C} \text{ iff } \neg A \text{ is unsatisfiable in } \mathcal{C}}$

Examples

- Singleton models: $\{M : card(W) = 1\}$
  valid: $\Diamond A \to \Box A$
- Reflexive models: KT
  valid: $\Box A \to A$
- Transitive models: K4
  valid: $\Diamond\Diamond A \to \Diamond A$
- Reflexive and transitive models: S4
  valid: ...
- Equivalence relation: S5
  valid: $A \to \Box\Diamond A$, ...
- ...

Reasoning problems

- Model checking
  Given M, w, and A do we have $M, w \Vdash A$?

- Validity
  Given A and $\mathcal{C}$ is A valid in $\mathcal{C}$?

- Satisfiability
  Given A and $\mathcal{C}$ does there exist M in $\mathcal{C}$ and w in M: $M, w \Vdash A$?

- Model construction
  Given A and $\mathcal{C}$ compute M in $\mathcal{C}$ and w in M: $M, w \Vdash A$

How can we solve them automatically?

Outline

1 Modal logics
- Possible worlds models
- Classes of models
- Language
- Semantics

2 Reasoning problems
- Validity and satisfiability in a class of models
- Outline of the tableaux method

Classical logic [Beth]

Checking the satisfiability of a given formula A:

1. Try to find M and w by applying truth conditions
   - $M, w \Vdash A_1 \wedge A_2 \implies$ add $M, w \Vdash A_1$, and add $M, w \Vdash A_2$
   - $M, w \Vdash A_1 \vee A_2 \implies$ either add $M, w \Vdash A_1$, or add $M, w \Vdash A_2$ (nondeterministic)
   - $M, w \Vdash \neg A_1 \implies$ don't add $M, w \Vdash A_1$!!
     - $M, w \Vdash \neg\neg A_1 \implies$ add $M, w \Vdash A_1$
     - $M, w \Vdash \neg(A_1 \vee A_2) \implies$ add $M, w \Vdash \neg A_1$ and add $M, w \Vdash \neg A_2$
     - $M, w \Vdash \neg(A_1 \wedge A_2) \implies$ add $M, w \Vdash \neg A_1$ or add $M, w \Vdash \neg A_2$

   $\implies$ tableau rules
2. apply while possible (saturation)
3. is M a model?
   - NO if both $M, w \Vdash B$ and $M, w \Vdash \neg B$ (closed tableau)
   - ELSE M is a model for A (open tableau) $W = \{w\}$, $R = \emptyset$, $V(w) = \{P : M, w \Vdash P\}$

Classical logic [Beth]

Checking the satisfiability of a given formula A:

1. Try to find M and w by applying truth conditions
   - $M, w \Vdash A_1 \wedge A_2 \implies$ add $M, w \Vdash A_1$, and add $M, w \Vdash A_2$
   - $M, w \Vdash A_1 \vee A_2 \implies$ either add $M, w \Vdash A_1$, or add $M, w \Vdash A_2$ (nondeterministic)
   - $M, w \Vdash \neg A_1 \implies$ don't add $M, w \Vdash A_1$!!
     - $M, w \Vdash \neg\neg A_1 \implies$ add $M, w \Vdash A_1$
     - $M, w \Vdash \neg(A_1 \vee A_2) \implies$ add $M, w \Vdash \neg A_1$ and add $M, w \Vdash \neg A_2$
     - $M, w \Vdash \neg(A_1 \wedge A_2) \implies$ add $M, w \Vdash \neg A_1$ or add $M, w \Vdash \neg A_2$

   $\implies$ tableau rules
2. apply while possible (saturation)
3. is M a model?
   - NO if both $M, w \Vdash B$ and $M, w \Vdash \neg B$ (closed tableau)
   - ELSE M is a model for A (open tableau) $W = \{w\}$, $R = \emptyset$, $V(w) = \{P : M, w \Vdash P\}$

Modal logic [Fitting]

Basic cases

- $M, w \Vdash \Diamond A$
  $\Longrightarrow$ add some new node u, add wRu, add $M, u \Vdash A$
- $M, w \Vdash \Box A$
  $\Longrightarrow$ for all node u s.th. wRu, add $M, u \Vdash A$

Apply truth conditions = build a labeled graph

- create nodes
- add links
- add formulas to nodes

Example

a node with the input formula

[] P & <> Q & <> (R v ~ P)

Example

$$M, w \Vdash A \wedge B \quad \text{iff} \quad M, w \Vdash A \quad \text{and} \quad M, w \Vdash B$$

$$\begin{array}{lll} A & \text{is} & \Box P \\ B & \text{is} & \Diamond Q \wedge \Diamond(R \vee \neg P) \end{array}$$

[] P & <> Q & <> (R v ~ P)

Example

$$M, w \Vdash A \wedge B \quad \text{iff} \quad M, w \Vdash A \quad \text{and} \quad M, w \Vdash B$$

$$\begin{array}{lll} A & \text{is} & \Box P \\ B & \text{is} & \Diamond Q \wedge \Diamond(R \vee \neg P) \end{array}$$

[] P & <> Q & <> (R v ~ P)
[] P
<> Q & <> (R v ~ P)

Example

$$M, w \Vdash A \wedge B \quad \text{iff} \quad M, w \Vdash A \quad \text{and} \quad M, w \Vdash B$$

[] P & <> Q & <> (R v ~ P)

[] P

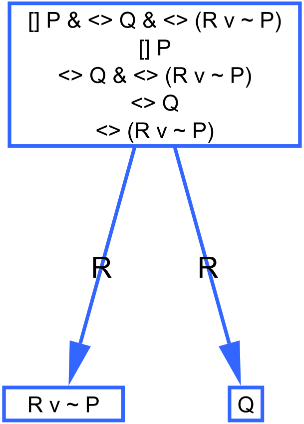
<> Q & <> (R v ~ P)

<> Q

<> (R v ~ P)

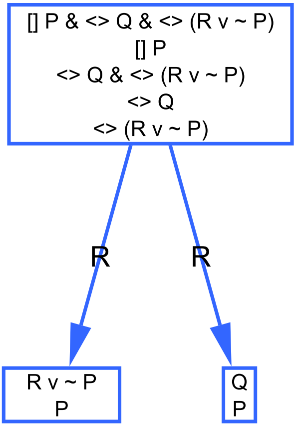
Example

$$M, w \Vdash \Diamond A \quad \text{iff} \quad \text{there is } u \text{ s.th. } wRu \quad \text{and} \quad M, u \Vdash A$$



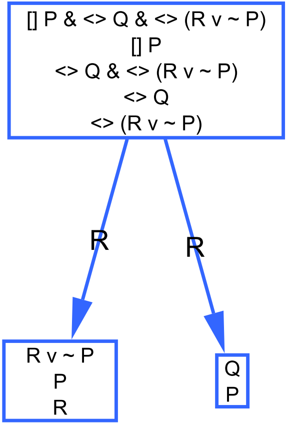
Example

$$M, w \Vdash \Box A \quad \text{iff} \quad \text{for all } u\text{: if } wRu \text{ then } M, u \Vdash A$$

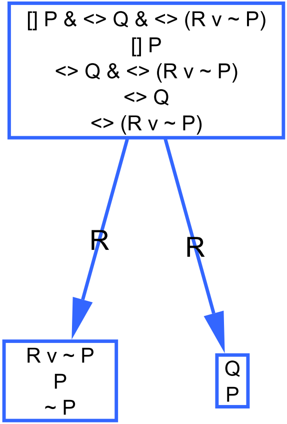


Example

$$M, w \Vdash A \vee B \quad \text{iff} \quad M, w \Vdash A \quad \text{or} \quad M, w \Vdash B$$

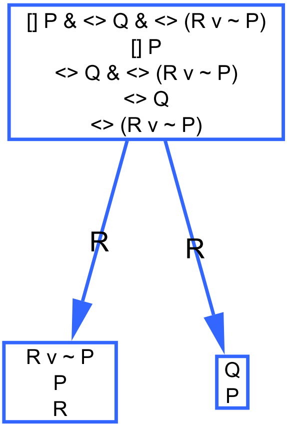


premodel 1

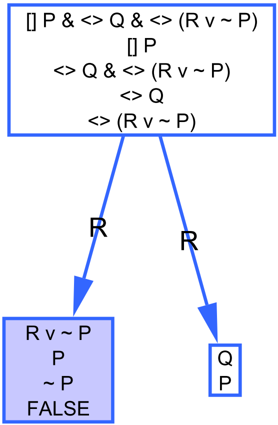


premodel 2

Example

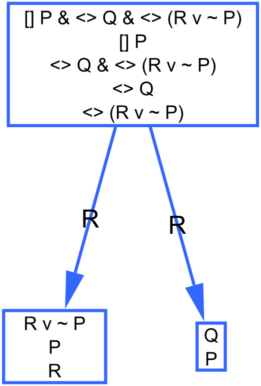


premodel 1



premodel 2

Example

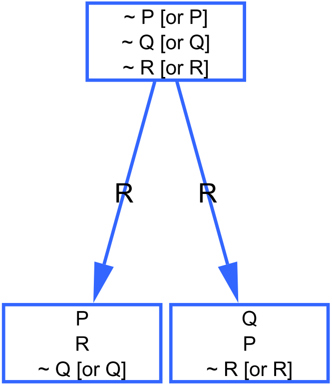


premodel 1

$\overline{\text{Model}}$

$\Longrightarrow$

$\underline{\text{extraction}}$



$M, w \Vdash P$ then $P \in V(w)$

A short history of tableaux

Handwritten proofs since 1950's

- . . . Sequent calculi [Beth, Gentzen]
- Tableaux calculi
  (tableau proof = sequent proof backwards)
- Kripke: explicit accessibility relation
- Smullyan, Fitting: uniform notation
- Single-step tableaux [Massacci]
  $\sigma : \Diamond A \implies \sigma, n : A$
- Tableaux by graph rewriting [Castilho et al.]

Nowadays: automated provers

- fast: FaCT [Horrocks], LWB [Heuerding, Jäger et col.], K-SAT [Giunchiglia&Sebastiani]
- generic: TWB [Abate&Goré], LoTREC

Part 2: Practice

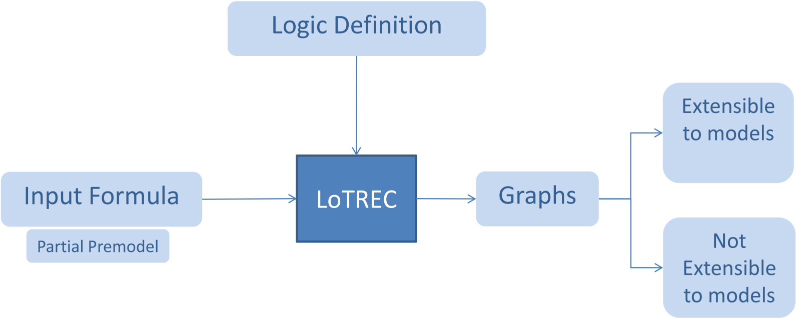
3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

A short history of LoTREC

- before 2000: theoretical bases (Luis Fariñas del Cerro, Olivier Gasquet, Andreas Herzig)
- David Fauthoux [2000]
  - rewriting kernel
  - event-based implementation
  - K, KT, KB
- Mohamad Sahade [2002-2005]
  - loopchecking
  - more logics: S4, K4, ...
  - general completeness and termination proofs
- Bilal Saïd [2006-2010]
  - LTL, PDL...
  - Confluence & commutative patterns
  - Model checking
  - graph rewriting basis & their theoretical properties
  - GUI, full web accessibility, step-by-step run,...
  - ...

The black box



Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

User-defined language

Atomic propositions

- Any constant symbol = Capital_1st_letter_words

Formulas

- Prefix notation (but can be displayed in infix form)
- Priority and associativity to avoid printing parentheses

Example (definition)

name	arity	display
`not`	1	`~ _`
`and`	2	`_ & _`
...		
`nec`	1	`[] _`
`pos`	1	`<> _`
...		

Example (usage)

- `pos P`
  displayed: <> P
- `and not Q not P`
  displayed: ∼ Q & ∼ P

Outline

3 LoTREC

- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

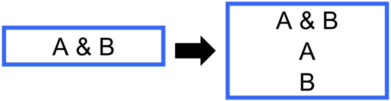
4 Implementing logics

- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

On paper

Truth conditions
$+$
Structural constraints **as** Graph rewriting rules

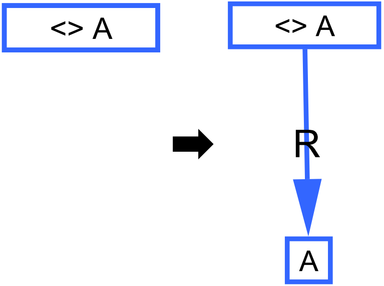
$M, w \Vdash A \wedge B$ iff
$M, w \Vdash A$ and $M, w \Vdash B$



On paper

Truth conditions
+
Structural constraints

as Graph rewriting rules



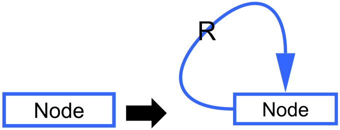
$M, w \Vdash \Diamond A$ iff
$\exists u$ s.th. wRu and
$M, u \Vdash A$

On paper

Truth conditions
+
Structural constraints

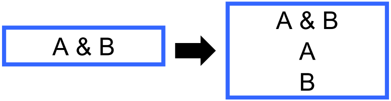
as Graph rewriting rules

Model is reflexive



In LoTREC

Graph rewriting rule as "if Conditions ... then Actions"

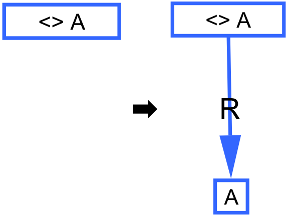



```
Rule And
 hasElement node and variable A variable B

 add node variable A
 add node variable B
End
```

In LoTREC

Graph rewriting rule as "if Conditions … then Actions"

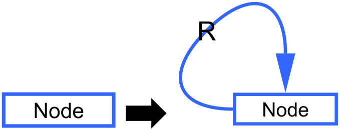



```
Rule Pos
  hasElement node1 pos variable A

  createNewNode node2
  link node1 node2 R
  add node2 variable A
End
```

In LoTREC

Graph rewriting rule as "if Conditions ... then Actions"




```
Rule ReflexiveEdges
 isNewNode node

 link node node R
End
```

Semantics of rules: the basic idea

Apply rule to a graph G = apply to every formula in every node
$\Longrightarrow$ strategies get more declarative
$\Longrightarrow$ proofs get easier

Tableau rules expand directed graphs by

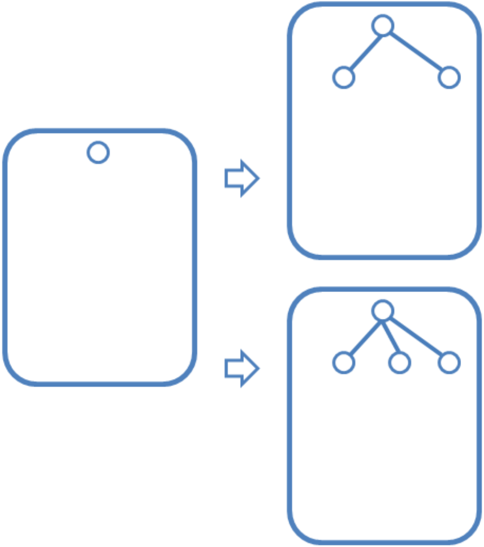
- adding links
- adding nodes
- adding formulas
- duplicating the graph

$$rule(G) = \{G_1, \ldots, G_n\}$$

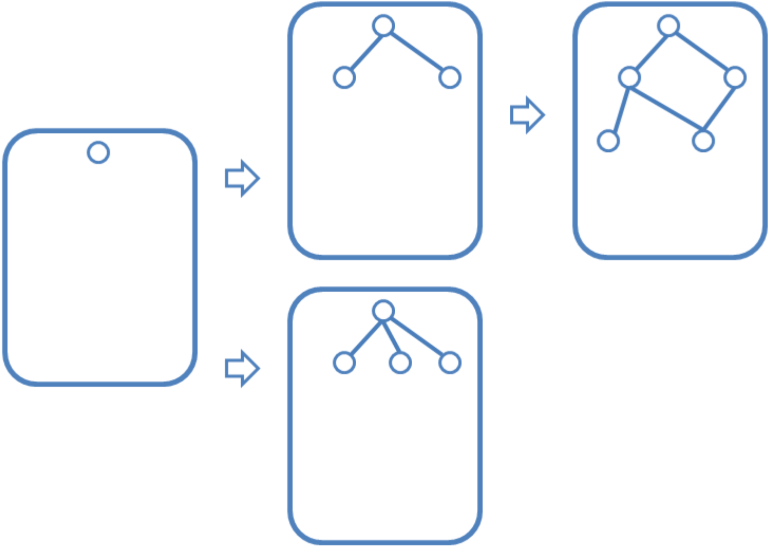
$$rule(\{G_1, \ldots, G_n\}) = rule(G_1) \cup \ldots \cup rule(G_n)$$

Managing graph copies: depth-first

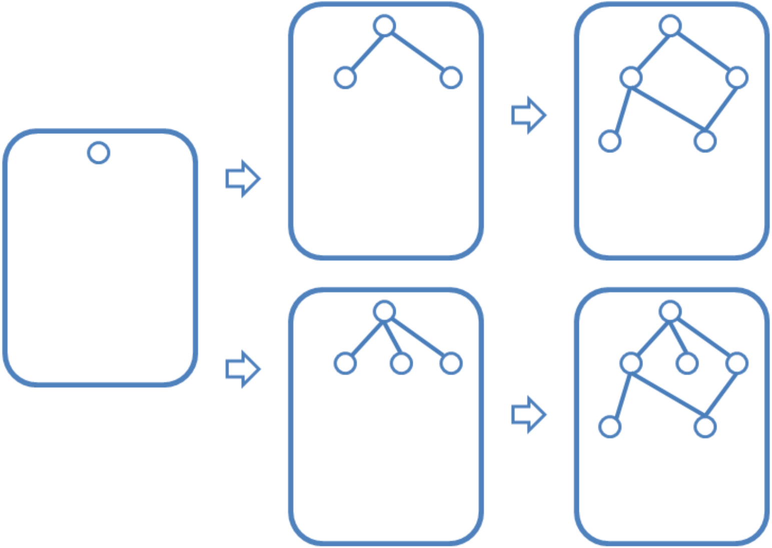
Managing graph copies: depth-first



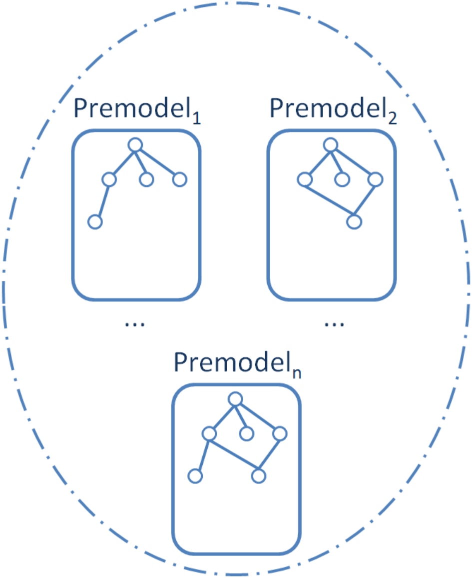
Managing graph copies: depth-first



Managing graph copies: depth-first



Managing graph copies: depth-first



Outline

3 LoTREC

- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics

- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

Why a strategy?

- Apply rules in order:

```
Strategy performOnce
 Stop
 And
 Or
 ...
```

- Saturation:

```
Strategy CPL_strat
 repeat
  Stop
  NotNot
   And
   Or
 end
```

```
Strategy K_strat
 repeat
  CPL
  Pos
  Nec
 end
```

Semantics of strategies

- block: `rule1 ... rulen ... anotherStrategy ...`
  apply all applicable rules in order then stop

Example

```
Strategy CPL
 Stop
 And
 Or
 Not_Not
 ...
```

Semantics of strategies

- block: `rule1 ... rulen ... anotherStrategy ...`
  apply all applicable rules in order then stop
- `repeat block end`
  repeat until no rule applicable (saturation)

Example

```
Strategy K
repeat
   CPL
   Pos
   Nec
end
```

For simple logics: `repeat` and blocks are sufficient!

Semantics of strategies

- block: `rule1 ... rulen ... anotherStrategy ...`
  apply all applicable rules in order then stop
- `repeat block end`
  repeat until no rule applicable (saturation)
- `firstRule block end`
  apply first applicable rule, then stop (unfair!) cf. higher-order proof assistants

Example

```
repeat
 firstRule
  rule1
  rule2    x
 end
end
```

`rule1` is always applicable
`rule2` is applicable
BUT never applied!

Semantics of strategies

- block: `rule1 ... rulen ... anotherStrategy ...`
  apply all applicable rules in order then stop
- `repeat block end`
  repeat until no rule applicable (saturation)
- `firstRule block end`
  apply first applicable rule, then stop (unfair!) cf. higher-order proof assistants
- `allRules block end`
  exactly as a "block", but needed inside `firstRule`

Example

```
firstRule
  rule1
  allRules
    rule2
    rule3
  end
  rule4
end
```

Semantics of strategies

- block: `rule1 ... rulen ... anotherStrategy ...`
  apply all applicable rules in order then stop
- `repeat block end`
  repeat until no rule applicable (saturation)
- `firstRule block end`
  apply first applicable rule, then stop (unfair!) cf. higher-order proof assistants
- `allRules block end`
  exactly as a "block", but needed inside `firstRule`
- `applyOnce rule`
  apply the rule on only one occurrence

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

Tableau definition

The set of tableaux for formula A with strategy S is the set of graphs obtained by applying the strategy S to an initial single-node graph whose root contains only A.

- Notation: $S(A)$

Remark

our tableau = "tableau branch" in the literature
(sounds odd to call a graph a branch)

Open or Closed?

- A node is closed iff it contains "FALSE" (unless. . .)
- A tableau is closed iff it has a closed node
- A set of tableaux is closed iff all its elements are closed

An open tableau is a premodel
$\Longrightarrow$ build a model

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

Formal properties

To be proved for each strategy S:

- Termination
  For every A, $S(A)$ terminates.
- Soundness
  If $S(A)$ is closed then A is unsatisfiable.
- Completeness
  If $S(A)$ is open then A is satisfiable.

In general. . .

- Soundness proofs: easy (we just apply truth conditions)
- Termination proofs: not so easy (case-by-case)
- Completeness proofs...
  - . . . for fair strategies: standard techniques
    work "in most cases"
    but fair strategies do not terminate in general
  - . . . for terminating strategies: difficult
    rigorous proofs rare even for the basic modal logics!
    reason: strategy = imperative programming

In general. . .

BUT soundness + termination is practically sufficient (e.g. when experimenting with a logic):

- given: class of models $\mathcal{C}$, strategy S, formula A
- apply strategy S to A
- take an open tableau and build pointed model (M, w)
- check if M in desired class of models
- check if $M, w \Vdash A$

A general termination theorem

[O. Gasquet et al., AIML 2006]

- IF for every rule ρ:
  the RHS of ρ contains **strict** subformulas of its LHS
  AND
  some restriction on node creation
- THEN
  for every formula A:
  the tableaux construction terminates

Another general termination theorem

[O. Gasquet et al., AIML 2006]

- IF for every rule ρ:
  the RHS of ρ contains subformulas of its LHS
  AND
  some restriction on node creation
  AND
  some loop testing in the strategy
- THEN
  for every formula A:
  the tableaux construction terminates

Part 2: Practice

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

How to get LoTREC

- or, *Download* $\Longrightarrow$ *Executable* to get *LoTREC_2.0.zip*
  - unzip
  - run file *run.bat*

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

How to proceed

CPL: Classical Propositional Logic

1. From the task pane, open:
   Open Predefined logic $\Longrightarrow$ *Others* $\Longrightarrow$ *CPL*
2. Run with
   Build Models
3. Why these results?
   - Predefined formula
   - Predefined Main strategy
4. Review the logic definition: *Connectors*, *Rules*...
5. Change the formula
6. Re-run...

Adding "$\leftrightarrow$"

What about formulas with "$\leftrightarrow$" connector?

1. Save as *CPL* locally as "*CPL_complete.xml*"
2. Add to *Connectors*:

name	arity	display	priority
`equiv`	2	`_<->_`	0 (lowest)

3. Add to *Rules*:
   `Equiv`, and `NotEquiv`
4. Call them in the strategy
5. Try some formulas. . .

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

From CPL to K

- Here: minimal set of connectors $\neg$, $\wedge$, $\Box$ only
- Rules of CPL
- Rule for $\neg\Box A$:
  - for every $\neg\Box A$ at every node w:
    create a successor u and add $\neg A$ to it
- Rule for $\Box A$:
  - for every $\Box A$ at every w, and for every R-successor u of w:
    add A to u
- Strategy: saturate with all the rules...

Rules

- Rule NotNec

  ```
  hasElement w pos variable a
  createNewNode u
  link w u R
  add u variable a
  ```

- Rule Nec

  ```
  hasElement w nec variable a
  isLinked w u R
  add u variable a
  ```

Strategies

1. Continue with your "*CPL_complete.xml*",
   or
   Open Predefined logic $\Longrightarrow$ *Others* $\Longrightarrow$ *CPL_complete*
2. Add the `nec` connector
3. Add the rules `Nec` and `NotNec`
4. Add a new strategy `KStrategy` which calls repeatedly `CPLStrategy` and then the rules `Pos` and `Nec`
5. Test with [] P & <> Q & <> (R v ∼ P)
   i.e. `and nec P and pos Q pos or R not P`
6. Test with other formulas...

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

From K To K_n

- Replace the connector $\Box_{\_}$ by $[\_]_{\_}$
- Change all the predefined formulae
- Change the modal rules: `Nec` and `NotNec`

```
Rule Nec_K
  hasElement w nec variable a
  isLinked w u R

  add u variable a
```

```
Rule Nec_Multimodal_K
  hasElement w nec variable r
                   variable a
  isLinked w u variable r

  add u variable a
```

How to proceed

1. From the task pane, open:
   Open Predefined logic $\Longrightarrow$ *Others* $\Longrightarrow$ *Multimodal-K*
2. Check $\neg[1]P \wedge \neg[2]\neg P, \ldots$

Description logic ALC

- Notational variant:
  - write R instead of I ('atomic role')
  - write A instead of P ('atomic concept')
  - write C instead of A ('complex concept')
  - write $\sqcap$ instead of $\wedge$
  - write $\sqcup$ instead of $\vee$
  - write $\forall R.C$ instead of $[I]A$
  - write $\exists R.C$ instead of $\langle I \rangle A$
- In LoTREC: change connectors and rules appropriately
- Test concept satisfiability:
  $\exists R.(A \sqcap A') \sqcap \forall R.\neg A$

  ...
- Test concept inclusion:
  $C_1 \sqsubseteq C_2$ iff $C_1 \sqcap \neg C_2$ unsatisfiable

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

From K to KT

Accessibility relation R is reflexive

- Aim: close all tableaux for $\Box P \land \neg P$ (negation of axiom T)
- Idea$_1$: integrate reflexivity into the truth condition
  - $M, w \Vdash \Box A$ iff $M, w \Vdash A$, and $M, u \Vdash A$ for every u that is accessible from w via R
- Idea$_2$: explicitly add reflexive edges to the graphs

From K to KT, ctd.

1. Save *Monomodal-K* as *Monomodal-KT*
2. Idea_1: add new rule

   ```
   Rule NecT
     hasElement w nec variable a
     add w variable a
   ```
3. Idea_2: add new rule

   ```
   Rule Reflexive_edges_for_R
     isNewNode w
     link w w R
   ```
4. Call new rule in the strategy
5. Check $P \wedge \Box\neg P$, $P \wedge \Box\Box\neg P$, ...

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

From K to KD

Accessibility relation R is serial

- Aim: close all tableaux for $\Box P \land \Box \neg P$ (negation of axiom D)
- Naive idea: just add edges

```
Rule makeSerial
    isNewNode w                    (match a node)
    createNewNode u
    link w u R
```

$\Longrightarrow$ will loop

From K to KD, ctd.

Accessibility relation R is serial

- Idea: add edges only when needed and not created elsewhere

```
Rule makeSerial
    hasElement w nec variable a
    hasNotElement w not nec variable b

    createNewNode u
    link w u R
```

- Call rule `makeSerial` in the strategy
- Check $\Box P \wedge \Box \neg P \ldots \Longrightarrow$ sound but suboptimal
- avoid too many successor nodes: apply `makeSerial` only once

```
    applyOnce makeSerial
```

From K to KD, ctd.

Accessibility relation R is serial

- Idea: add edges only when needed and not created elsewhere

```
Rule makeSerial
   hasElement w nec variable a
   hasNotElement w not nec variable b

   createNewNode u
   link w u R
```

- Call rule `makeSerial` in the strategy
- Check $\Box P \wedge \Box\neg P \ldots \Longrightarrow$ sound but suboptimal
- avoid too many successor nodes: apply `makeSerial` only once

```
applyOnce makeSerial
```

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

From KT to S4

- Accessibility relation R is reflexive and transitive (S4 = KT4)
- Aim: close all tableaux for $\Box P \wedge \neg\Box\Box P$

  (negation of axiom 4)
- Idea$_1$: integrate reflexivity and transitivity into the truth condition
  - $M, w \Vdash \Box A$ iff $M, w \Vdash A$, and $M, u \Vdash \Box A$ for every u that is accessible from w via R
- Idea$_2$: ...

From KT to S4, ctd.

1. Save *Monomodal-KT* as *Monomodal-S4*
2. Copy/Paste rule `Nec`, and rename it as `Nec4`
3. Idea$_1$:

```
Rule Nec4
  hasElement node nec R variable a
  isLinked node node' R

  add node' nec R variable a
```

4. Check $\neg(\Box P \rightarrow \Box\Box P)$, i.e. $\Box P \wedge \neg\Box\Box P$
5. Test $\Box\neg\Box P$

Taming S4

- LoTREC loops on input formula $\Box\neg\Box P$!
- Execute step-by-step ('Step By Step' instead of 'Build Premodels' button)
- Observe: if no clash wasn't found after 2 nodes, there is no chance to find it later
  $\Longrightarrow$ *no need to create successors for nodes that are included in an ancestor!*
  - hypothesis: nodes have been locally saturated before checking for loops

Taming S4, ctd.

- Add the rule `loopTest` (cf. predefined S4_Optimal)

```
Rule loopTest
  isNewNode node'                    (required for local activation)
  isAncestor node node'
  contains node node'

  mark node' CONTAINED
  link node' node Loop               (optional, highlights the inclusion)
```

- add condition to rule `NotNec`:

```
  hasElement node not nec A
  isNotMarked node CONTAINED
  ...
```

- Call rule `loopTest` in the strategy
  - guarantee that nodes are saturated before loopchecking: call `loopTest` after the CPL rules and rule NecT
- Run again...

Taming S4, ctd.

- Add the rule `loopTest` (cf. predefined S4_Optimal)

```
Rule loopTest
  isNewNode node'                (required for local activation)
  isAncestor node node'
  contains node node'
  mark node' CONTAINED
  link node' node Loop           (optional, highlights the inclusion)
```

- add condition to rule `NotNec`:

```
  hasElement node not nec A
  isNotMarked node CONTAINED
  ...
```

- Call rule `loopTest` in the strategy
  - guarantee that nodes are saturated before loopchecking: call `loopTest` after the CPL rules and rule NecT
- Run again. . .

Taming S4, ctd.

- Add the rule `loopTest` (cf. predefined S4_Optimal)

```
Rule loopTest
  isNewNode node'                    (required for local activation)
  isAncestor node node'
  contains node node'
  mark node' CONTAINED
  link node' node Loop               (optional, highlights the inclusion)
```

- add condition to rule `NotNec`:

```
  hasElement node not nec A
  isNotMarked node CONTAINED
  ...
```

- Call rule `loopTest` in the strategy
  - guarantee that nodes are saturated before loopchecking: call `loopTest` after the CPL rules and rule NecT
- Run again. . .

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

From S4 to intuitionistic logic LJ

- Accessibility relation R is reflexive, transitive, and *hereditary*
- Truth conditions:
  $M, w \Vdash A \rightarrow B$ iff $M, u \nVdash A$ or $M, u \Vdash B$ for all u s.th. wRu
  $M, w \Vdash \neg A$ iff $M, u \nVdash A$ for all u s.th. wRu
- not valid: $\neg\neg A \leftrightarrow A$; $\neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$; ...
  - tableau method requires *signed formulas*
  - in LoTREC: define connectors sTrue and sFalse
- Rules for conjunction:

```
Rule sTrueAnd
   hasElement w sTrue and variable a variable b
   add w sTrue variable a
   add w sTrue variable b
Rule sFalseAnd
   hasElement w sFalse and variable a variable b
   duplicate copiedgraph
   add w sFalse variable a
   add copiedgraph.w sFalse variable b
```

From S4 to intuitionistic logic LJ

- Accessibility relation R is reflexive, transitive, and *hereditary*
- Truth conditions:
  $M, w \Vdash A \to B$ iff $M, u \not\Vdash A$ or $M, u \Vdash B$ for all u s.th. wRu
  $M, w \Vdash \neg A$ iff $M, u \not\Vdash A$ for all u s.th. wRu
- not valid: $\neg\neg A \leftrightarrow A$; $\neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$; ...
  - tableau method requires *signed formulas*
  - in LoTREC: define connectors `sTrue` and `sFalse`
- Rules for conjunction:

```
Rule sTrueAnd
   hasElement w sTrue and variable a variable b
   add w sTrue variable a
   add w sTrue variable b
Rule sFalseAnd
   hasElement w sFalse and variable a variable b
   duplicate copiedgraph
   add w sFalse variable a
   add copiedgraph.w sFalse variable b
```

From S4 to intuitionistic logic LJ

- Accessibility relation R is reflexive, transitive, and *hereditary*
- Truth conditions:
  $M, w \Vdash A \to B$ iff $M, u \nVdash A$ or $M, u \Vdash B$ for all u s.th. wRu
  $M, w \Vdash \neg A$ iff $M, u \nVdash A$ for all u s.th. wRu
- not valid: $\neg\neg A \leftrightarrow A$; $\neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$; ...
  - tableau method requires *signed formulas*
  - in LoTREC: define connectors `sTrue` and `sFalse`
- Rules for conjunction:

```
Rule sTrueAnd
   hasElement w sTrue and variable a variable b
   add w sTrue variable a
   add w sTrue variable b
Rule sFalseAnd
   hasElement w sFalse and variable a variable b
   duplicate copiedgraph
   add w sFalse variable a
   add copiedgraph.w sFalse variable b
```

From S4 to intuitionistic logic LJ, ctd.

- Rules for implication:

```
Rule sFalseImp
   hasElement w sFalse imp variable a variable b
   isNotMarked w CONTAINED
   createNewNode u
   link w u R
   add u sTrue variable a
   add u sFalse variable b
Rule sTrueImpActual
   hasElement w sTrue imp variable a variable b
   add w sFalse variable a
   add copiedgraph.w sTrue variable b
   duplicate copiedgraph
Rule sTrueImpPropagation
   hasElement w sTrue imp variable a variable b
   isLinked w u R
```

From S4 to intuitionistic logic LJ, ctd.

- Rule for true atoms (implements hereditary R):

```
Rule sTrueAtom
   hasElement w sTrue variable a
   isAtomic variable a
   isLinked w u R

   add u sTrue variable a
```

- Test:
  $((P \to Q) \to P) \to P$ (Pierce's formula)
- Test:
  $\neg\neg P \to P$
  $P \to \neg\neg P$
  $P \vee \neg P$
  ...
- improve: use three signs...

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

Model checking

Given M_0, w_0, and A_0... do we have $M_0, w_0 \Vdash A_0$?

1. build model M_0 with root w_0 in LoTREC

   ```
   createNewNode w0,
   createNewNode u,
   link w0 u R,
   add u P,
   add u Q,
   ...
   ```

2. add formula A_0 to be checked to root note w_0

   `add w0 isItTrue nec not P` (add as dummy connector)

3. top-down: decomposition of A_0

   ```
   hasElement w isItTrue not variable A
   add w isItTrue variable A

   hasElement w isItTrue nec variable A
   isLinked w u R
   add u isItTrue variable A
   ...
   ```

Model checking, ctd.

4. bottom-up: build truth value of A_0

```
hasElement w isItTrue variable A
isAtomic variable A
hasElement w variable A
markExpression w isItTrue variable A Yes

hasElement w isItTrue nec variable A
isLinked w u R
isMarkedExpression u isItTrue variable A No
markExpression w isItTrue nec variable A No

hasElement w isItTrue nec variable A
isLinked w u R
isMarkedExpressionInAllChildren w isItTrue variable A R Yes

markExpression w isItTrue nec variable A Yes
```

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

Propositional Dynamic Logic PDL

- Language: complex programs Π, complex formulas A

$$\Pi ::= I \mid A? \mid \Pi;\Pi \mid \Pi \cup \Pi \mid \Pi^*$$

$$A ::= P \mid \neg A \mid A \wedge A \mid A \vee A \mid \langle \Pi \rangle A \mid [\Pi]A$$

  where P ranges over $\mathcal{P}$ and I ranges over $\mathcal{I}$
- Interpretation of complex programs and formulas: defined by mutual recursion
  - $R_{A?} = \{\langle w, w\rangle : M, w \Vdash A\}$
  - $R_{\Pi_1;\Pi_2} = R_{\Pi_1} \circ R_{\Pi_2}$
  - $R_{\Pi_1 \cup \Pi_2} = R_{\Pi_1} \cup R_{\Pi_2}$
  - $R_{\Pi^*} = (R_\Pi)^*$
  - $M, w \Vdash \langle \Pi \rangle A$ iff there is w' such that $wR_\Pi w'$ and $M, w' \Vdash A$

PDL: taming the Kleene star

- Problem: how to handle transitive closure?
- Solution: postpone
  - $M, w \Vdash [\Pi^*]A$ iff $M, w \Vdash A \wedge [\Pi][\Pi^*]A$
- in LoTREC:

```
Rule Nec_Star
 hasElement w nec star variable Pi variable A
 add w variable A
 add w nec variable Pi nec star variable Pi
                                          variable A

Rule Pos_Star
 hasElement w pos star variable Pi variable A
 add w or variable A pos variable Pi pos  ...
```

- termination: use looptesting
  - Observe: these rules don't add subformulas
  - ... but 'almost' subformulas (Fischer-Ladner closure)

PDL: taming the Kleene star

- Problem: how to handle transitive closure?
- Solution: postpone
  - $M, w \Vdash [\Pi^*]A$ iff $M, w \Vdash A \wedge [\Pi][\Pi^*]A$
- in LoTREC:

```
Rule Nec_Star
 hasElement w nec star variable Pi variable A
 add w variable A
 add w nec variable Pi nec star variable Pi
                                            variable A

Rule Pos_Star
 hasElement w pos star variable Pi variable A
 add w or variable A pos variable Pi pos  ...
```

- termination: use looptesting
  - Observe: these rules don't add subformulas
  - ... but 'almost' subformulas (Fischer-Ladner closure)

PDL: taming the Kleene star, ctd.

- A problem:
  - execute $\langle I^* \rangle P$ step-by-step
  - always choose the graph where the fulfillment of $\langle I^* \rangle P$ is postponed
  - observe: terminates by looptest, but $\langle I^* \rangle P$ not fulfilled $\Longrightarrow$ premodel cannot be transformed into a model of $\langle I^* \rangle P$
- Solution: check whether are all eventualities are fulfilled $\Longrightarrow$ use model checking, v.s.

Outline

3 LoTREC
- Language
- Rules
- Strategies
- Tableau notation
- Do the algorithms do the right thing?

4 Implementing logics
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

It is up to you...

- S5; K +Universal operator
- Confluence
- LTL
- ...

Thank you!