

Kripke's World

An introduction to modal logics via tableau systems

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

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 (\implies 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

Early history: les tableaux de Monsieur Toulouse-Lautrec



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

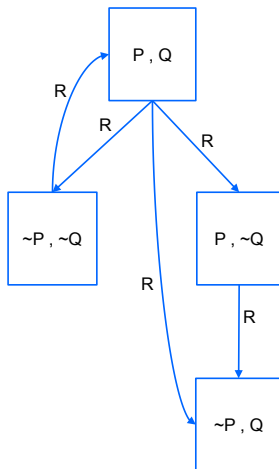
- Possible worlds models
- Classes of models
- Language
- Semantics

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

- $R: \mathcal{I} \longrightarrow 2^{W \times W}$

- $V: W \longrightarrow 2^{\mathcal{P}}$

set of possible worlds
accessibility relation
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_lu iff u is possible for agent l at actual world w
- Deontic:
 wRu iff u is an ideal counterpart of the actual world w
- Dynamic:
 wR_lu iff u is a possible result of the execution of the program / action l in w

Readings of $R \implies$ 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:
 $wR_l u$ iff u is possible for agent l at actual world w
- Deontic:
 wRu iff u is an ideal counterpart of the actual world w
- Dynamic:
 $wR_l u$ iff u is a possible result of the execution of the program / action l in w

Readings of $R \implies$ 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:
 $\{M : \text{card}(W) = 1\}$
- 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 $w = u$
 \implies hybrid logic
- R is hereditary (atomic propositions persist)
if $P \in V(w)$ and wRu then $P \in V(u)$
 \implies intuitionistic logic

Closing under constraints in LoTREC

- Closing under reflexivity:
condition: `isNewNode w`
action: `link w w R`
- Observe:
capital first letter \implies constant
small first letter \implies 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

$\neg A$	=	“not A ”
$A \wedge B$	=	“ A and B ”
$A \vee B$	=	“ A or B ”
$A \rightarrow B$	=	“if A then B ”
$A \leftrightarrow B$	=	“ A if and only if B ”
$A \oplus B$	=	“either A or B ”
$\oplus(A, B, C)$	=	“either A , or B , or C ”
		...

Modal formulas

■ Temporal logic

XA = “A will be true at the *next* time point”

FA = “A will be true at *some* time point in the future”
= “A will eventually be true”

GA = “A will be true at *every* time point in the future”
= “A will be true henceforth”

AUB = “A until B”

ASB = “A since B”

■ Dynamic logic

$\text{After}_I A$ = “A will be true after *every possible* execution of program *I*”

$\text{Feasible}_I A$ = “A will be true after *some* execution of program *I*”

(programs may be nondeterministic)

Modal formulas (ctd.)

- Epistemic and doxastic logic

$\text{Bel}_I A$ = “agent I believes that A ”

$\text{K}_I A$ = “agent I knows that A ”

$\hat{\text{Bel}}_I A$ = “it is (doxastically) possible for agent I that A ”

$\hat{\text{K}}_I A$ = “it is (epistemically) possible for agent I that A ”

- Deontic logic

$\text{O}_I B$ = “ A is obligatory for I ”

$\text{P}_I B$ = “ 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

$$\diamond A = \text{MA} = \text{“A is possible”}$$

$$\square A = \text{LA} = \text{“A is necessary”}$$

- Multimodal version:

$$\diamond_I A = \langle I \rangle A = \text{“A is possible w.r.t. } I\text{”}$$

$$\square_I A = [I] A = \dots$$

where $I \in \mathcal{I}$ (set of parameters)

- Common feature: Not truth-functional

- no f s.th. $\text{truthvalue}(\diamond A) = f(\text{truthvalue}(A))$

Duality

- Intuitively:

$$\hat{K}_I A \leftrightarrow \neg K_I \neg A$$

$$P_I A \leftrightarrow \neg O_I \neg A$$

$$FA \leftrightarrow \neg G \neg A$$

$$\text{After}_I A \leftrightarrow \neg \text{Feasible}_I \neg A$$

...

- Abstracting:

$$\diamond A \leftrightarrow \neg \square \neg A$$

$$\square A \leftrightarrow \neg \diamond \neg A$$

- Options:

- take both \diamond and \square as primitive

- take \diamond as primitive, and set $\square A \stackrel{\text{def}}{=} \neg \diamond \neg A$

- take \square as primitive, and set $\diamond A \stackrel{\text{def}}{=} \neg \square \neg A$

How define a language?

- Examples

- $CardRed \wedge K_{Ann}CardRed \wedge K_{Ann}\neg K_{Bob}CardRed$
- $DoorClosed \wedge [Open]DoorOpen$
- $P \wedge \neg Q \wedge \square Q \wedge \diamond(P \wedge \square\neg Q)$

- Language = set of formulas

- Language is defined by BNF:

$$A ::= P \mid \neg A \mid A \wedge A \mid A \vee A \mid \diamond A \mid \square A \mid \langle I \rangle A \mid [I]A \mid K_I A \mid \dots$$

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

How define a language in LoTREC?

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

$$\neg A = \text{not}(A)$$

$$A \wedge B = \text{and}(A, B)$$

$$A \vee B = \text{or}(A, B)$$

...

$$\text{Bel}_I A = \text{Bel}(I, A)$$

$$\text{K}_I A = \text{Knows}(I, A)$$

$$\hat{\text{K}}_I A = \text{Poss}(I, A)$$

...

$$A \text{U} B = \text{Until}(A, B)$$

...

$$A \Rightarrow B = \text{ifThen}(A, B)$$

...

- A LoTREC formula is
 - a propositional variable $P \in \mathcal{P}$, or
 - an expression of the form $op(A_1, \dots, A_n)$ where op is the name of a logical connector and the A_i are formulas or in \mathcal{I} \implies General schema: $op(\text{Arg}_1, \dots, \text{Arg}_n)$, where $\text{Arg}_i \in \mathcal{P} \cup \mathcal{I}$

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. $wR_I u$ and $M, u \Vdash A$
- ...

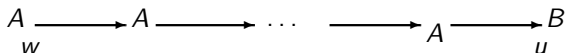
■ Relation algebra operators

- $M, w \Vdash \diamond^{-1} A$ iff there exists u s.th. $wR^{-1} u$ 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)

- $M, w \Vdash \mathbf{X}A$ iff there exists u s.th. wRu and $M, u \Vdash A$
- $M, w \Vdash \mathbf{F}A$ iff there exists n, u s.th. $wR^n u$ and $M, u \Vdash A$



- $M, w \Vdash \mathbf{A}U\mathbf{B}$ 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

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

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

- $\Box(P \vee \neg P)$
 - $\Box P \wedge \Box Q \rightarrow \Box(P \wedge Q)$
- 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 some class of models

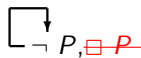
\mathcal{C} = some subset of \mathbf{K}

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

Example

- $\Box P \rightarrow P$ invalid in \mathbf{K} $\Box P, \neg P \longrightarrow P$

- $\Box P \rightarrow P$ valid in the class of reflexive models



- $\Diamond\Diamond P \rightarrow \Diamond P$ 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 \mathbf{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

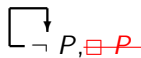
\mathcal{C} = some subset of \mathbf{K}

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

Example

- $\Box P \rightarrow P$ invalid in \mathbf{K} $\Box P, \neg P \longrightarrow P$

- $\Box P \rightarrow P$ valid in the class of reflexive models



- $\Diamond\Diamond P \rightarrow \Diamond P$ 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 \mathbf{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

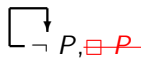
\mathcal{C} = some subset of \mathbf{K}

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

Example

- $\Box P \rightarrow P$ invalid in \mathbf{K} $\Box P, \neg P \longrightarrow P$

- $\Box P \rightarrow P$ valid in the class of reflexive models



- $\Diamond \Diamond P \rightarrow \Diamond P$ 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 \mathbf{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}

Examples

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

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

- $M, w \Vdash \Diamond A$
 \implies add some new node u , add wRu , add $M, u \Vdash A$
- $M, w \Vdash \Box A$
 \implies 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

$$\boxed{\neg P \ \& \ \leftrightarrow \ Q \ \& \ \leftrightarrow \ (R \vee \sim P)}$$

Example

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

A is $\Box P$

B is $\Diamond Q \wedge \Diamond(R \vee \neg P)$

$\Box P \ \& \ \Leftrightarrow \ \Diamond Q \ \& \ \Leftrightarrow \ (R \vee \sim P)$

Example

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

A is $\Box P$

B is $\Diamond Q \wedge \Diamond(R \vee \neg P)$

$\Box P \ \& \ \langle \rangle Q \ \& \ \langle \rangle (R \vee \sim P)$

$\Box P$

$\langle \rangle Q \ \& \ \langle \rangle (R \vee \sim P)$

Example

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

$\Box P \ \& \ \langle \rangle Q \ \& \ \langle \rangle (R \vee \sim P)$

$\Box P$

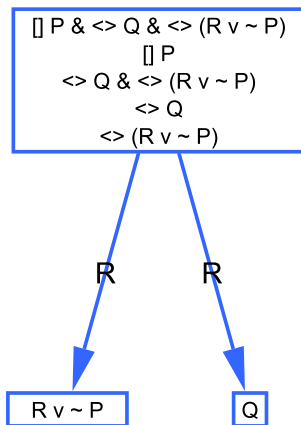
$\langle \rangle Q \ \& \ \langle \rangle (R \vee \sim P)$

$\langle \rangle Q$

$\langle \rangle (R \vee \sim P)$

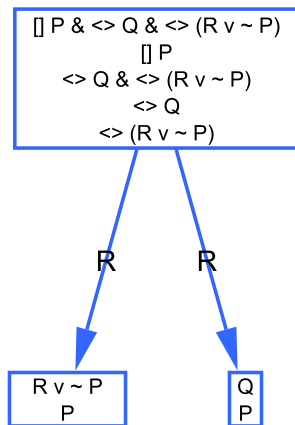
Example

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



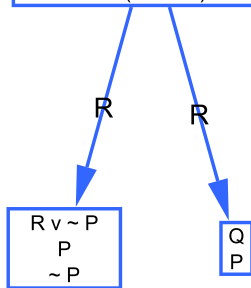
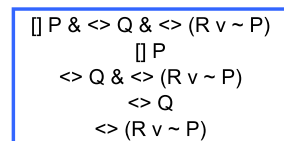
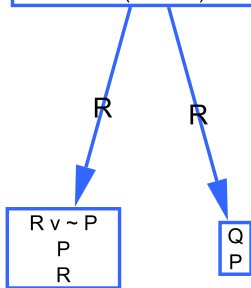
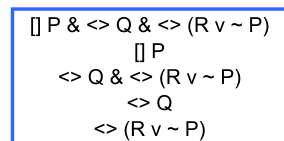
Example

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

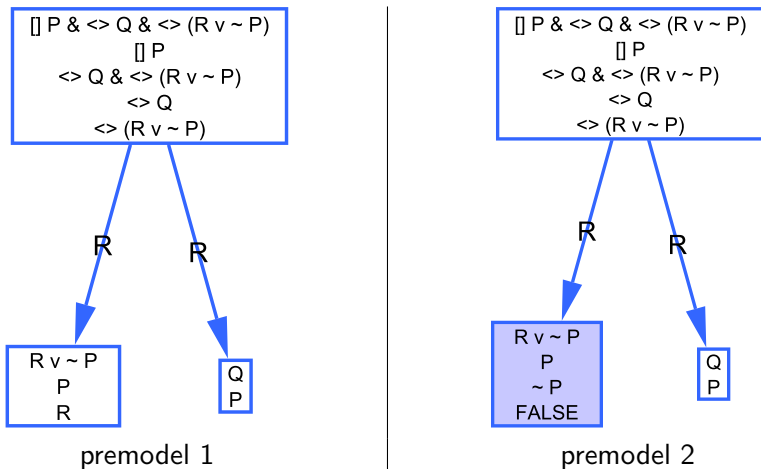


Example

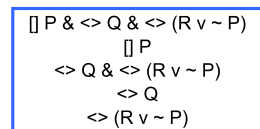
$M, w \Vdash A \vee B$ iff $M, w \Vdash A$ or $M, w \Vdash B$



Example

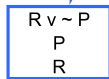


Example



R

R



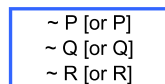
premodel 1



 Model

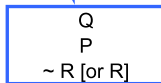
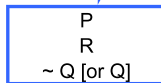
 \implies

 extraction



R

R


 $M, w \Vdash P$ 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?

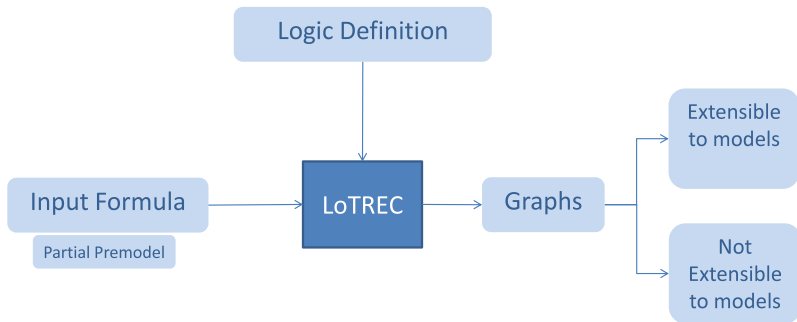
4 Implementing logics

- 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	\sim _
and	2	_ & _
...		
nec	1	[] _
pos	1	< > _
...		

Example (usage)

- pos P
displayed: $\langle \rangle P$
- and not Q not P
displayed: $\sim Q \& \sim P$

Outline

3 LoTREC

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

4 Implementing logics

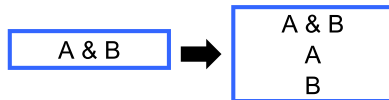
- Classical logic
- Modal logic K
- Multi-modal logic K_n
- KT
- KD
- S4
- Intuitionistic logic LJ
- Model checking in LoTREC
- PDL
- Suggestions

On paper

Truth conditions
+
Structural constraints

as Graph rewriting rules

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

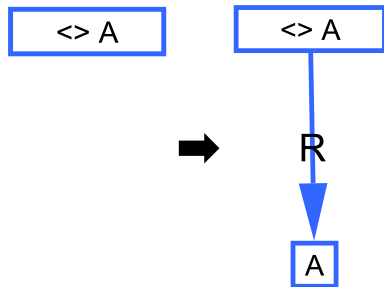


On paper

Truth conditions
+
Structural constraints

as Graph rewriting rules

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

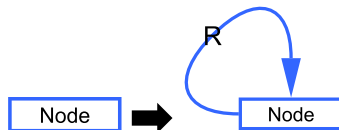


On paper

Truth conditions
+
Structural constraints

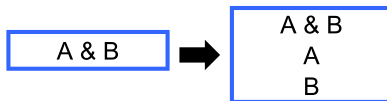
as Graph rewriting rules

Model is reflexive



In LoTREC

Graph rewriting rule as “if Conditions ... then Actions”



Rule And

hasElement node and variable A variable B

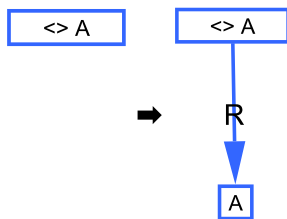
add node variable A

add node variable B

End

In LoTREC

Graph rewriting rule as “if Conditions ... then Actions”



Rule Pos

hasElement node1 pos variable A

createNewNode node2

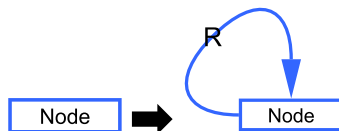
link node1 node2 R

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

\implies strategies get more declarative

\implies proofs get easier

Tableau rules expand directed graphs by

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

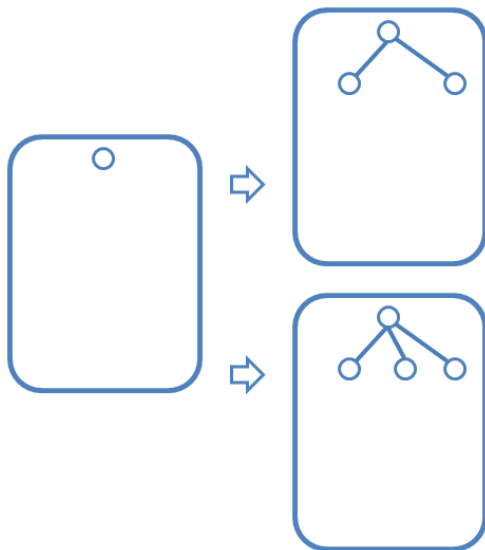
$$rule(G) = \{G_1, \dots, G_n\}$$

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

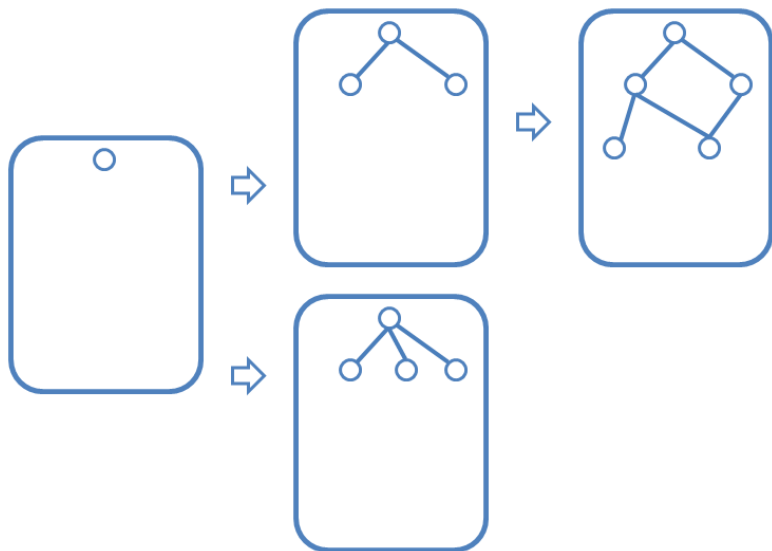
Managing graph copies: depth-first



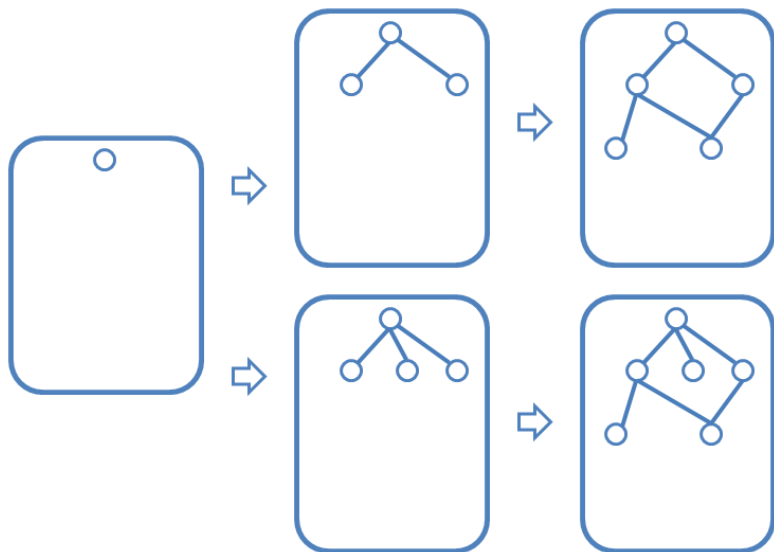
Managing graph copies: depth-first



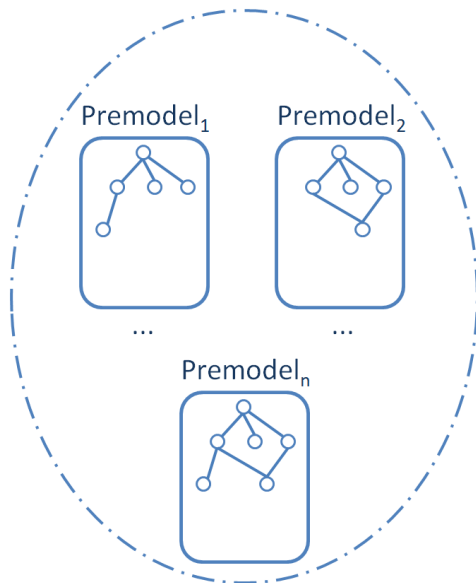
Managing graph copies: depth-first



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 ... rule_n ... 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
 \implies 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

`http://www.irit.fr/Lotrec` (Capital "L")

-  Webstart
- or, *Download* \implies *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 \implies Others \implies 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	\leftrightarrow	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 \implies Others \implies 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 \implies Others \implies Multimodal-K
- 2 Check $\neg[1]P \wedge \neg[2]\neg P, \dots$

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 $\neg(\Box P \rightarrow P)$ (negation of axiom T)
- Idea₁: 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
- Idea₂: explicitly add reflexive edges to the graphs

From K to KT, ctd.

1 Save *Monomodal-K* as *Monomodal-KT*

2 Idea₁: add new rule

Rule NecT

hasElement w nec variable a

add w variable a

3 Idea₂: 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 $\Box P \wedge \Box \neg P$ (negation of axiom D)
- Naive idea: just add edges

Rule makeSerial

isNewNode w (match a node)

createNewNode u

link w u R

\implies 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 $\Box P \wedge \Box \neg P \dots \implies$ 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 $\Box P \wedge \Box \neg P \dots \implies$ 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 $\neg(\Box P \rightarrow \Box\Box P)$
(negation of axiom 4)
- Idea₁: 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
- Idea₂: ...

From KT to S4, ctd.

- 1 Save *Monomodal-KT* as *Monomodal-S4*
- 2 Copy/Paste rule Nec, and rename it as Nec4
- 3 Idea₁:

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 $\Box\neg\Box P \dots$

Taming S4

- Input formula $\Box\neg\Box P$ loops!
- 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
 \implies *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 only here)

`isAncestor` `node node`'

`contains` `node node`' (or: `haveSameFormulaSet`)

`mark` `node`' `CONTAINED`

`link` `node`' `node Loop` (optional, marks the inclusion)

- add condition to rule `NotNec`: `isNotMarked` w `CONTAINED`
- Call it 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 LJ

Accessibility relation R is reflexive, transitive, and *hereditary*

- $M, w \Vdash A \rightarrow A$ iff for all u such that wRu , $M, u \not\Vdash A$ or $M, u \Vdash A$
- Tableau method requires *signed formulas*
 - in LoTREC: define connectors `mTrue` and `mFalse`
- Rules:

Rule `mFalseImp`

hasElement w `mFalse` `imp` variable a variable b
isNotMarked w CONTAINED

createNewNode u

link w u R

add u `mTrue` variable a

add u `mFalse` variable b

...

From S4 to LJ, ctd.

- Rule to propagate true atoms:

Rule mTrueAtom

hasElement w mTrue variable a

isAtomic variable a

isLinked w u R

add u mTrue variable a

- Test:

$$((P \rightarrow Q) \rightarrow P) \rightarrow P$$

(Pierce's formula)

- Add the other connectives...

- Test:

$$\neg\neg P \rightarrow P$$

$$P \rightarrow \neg\neg P$$

$$P \vee \neg P$$

...

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

Model checking

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

language: not, and, or, nec, pos

1. build model M in LoTREC

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

2. add formula A to be checked to root note

```
add w isItTrue nec P (to be added as dummy connector)
```

3. top-down: decomposition of A

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

4. bottom-up: build truth value of A ...

Model checking, ctd.

4. bottom-up: build truth value of A

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

PDL

program constructions: Kleene star, ...

- ...

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!