Kripke's World*s*

An introduction to modal logics via the tableau method

Olivier Gasquet, Andreas Herzig,
Bilal Said, François Schwarzentruber

Institut de Recherche en Informatique de Toulouse (IRIT)
Université de Toulouse

`http://www.irit.fr/~Andreas.Herzig/CTableaux`

ESSLLI 2010, Copenhagen

1st week, foundational course

Classical propositional logic CPL in a slide

- Language:
  - set $\mathcal{P}$ of propositional variables $P, Q, \ldots$
  - Boolean operators $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\uparrow$,...
  - (complex) formulas A, B,...
- Models:
  - valuations $V \subseteq \mathcal{P}$
- Semantics:
  - truth conditions:

    $V \Vdash A \rightarrow B$ iff $V \nVdash A$ or $V \Vdash B$

    ...
  - A is CPL-valid ($\models_{\mathrm{CPL}} A$) iff for every valuation V, $V \Vdash A$

    $\models_{\mathrm{CPL}} P \vee \neg P$ $\qquad$ $\models_{\mathrm{CPL}} ((P \rightarrow Q) \rightarrow P) \rightarrow P$

    $\models_{\mathrm{CPL}} \neg\neg P \rightarrow P$ $\qquad$ $\models_{\mathrm{CPL}} (P \rightarrow Q) \leftrightarrow (\neg P \vee Q)$

    $\models_{\mathrm{CPL}} P \rightarrow (Q \rightarrow P)$ $\qquad$ ...
  - A is CPL-satisfiable iff for some valuation V, $V \models A$

    P $\qquad$ ...

    $P \rightarrow Q \wedge \neg(Q \rightarrow P)$

First-Order Logic FOL in two slides

- Language:
  - object variables x, y,. . .
  - Predicates: $R(t_1, \dots, t_n)$
    - propositional variables = predicates of arity 0
    - a particular binary predicate: $equals(t_1, t_2)$, written $t_1 = t_2$
  - Complex formulas: built with CPL operators and $\forall x$, $\exists x$

$$\exists y \forall x R(x, y) \rightarrow \forall x \exists y R(x, y)$$

- Models:
  - domain D (must be non-empty)
  - interpretation of an n-ary predicate: $I(R) \subseteq D^n$
  - interpretation of a variable: $I(x) \in D$

$$I(equals) = \{\langle d, d\rangle \mid d \in D\}$$

First-Order Logic FOL in two slides (ctd.)

- Semantics:
  - truth conditions:

    $$\begin{array}{l} (D,I) \Vdash R(t_1,\dots,t_n) \text{ iff } \langle I(t_1),\dots,I(t_n)\rangle \in I(R) \\ (D,I) \Vdash \forall x A \text{ iff } (D,I') \Vdash A \text{ for all } x\text{-variants } I' \text{ of } I \\ (D,I) \Vdash \exists x A \text{ iff } (D,I') \Vdash A \text{ for some } x\text{-variant } I' \text{ of } I \end{array}$$

    where an x-variant of I interprets everything as I except for x

  - A is FOL-valid ($\models_{\mathsf{FOL}} A$) iff for every $\langle D,I\rangle$, $\langle D,I\rangle \Vdash A$

    $$\begin{array}{ll} \models_{\mathsf{FOL}} \forall x R(x) \to \exists x R(x) & \models_{\mathsf{FOL}} \exists y \forall x R(x,y) \to \forall x \exists y R(x,y) \\ \models_{\mathsf{FOL}} \exists x R(x) \leftrightarrow \neg\forall x \neg R(x) & \dots \end{array}$$

  - A is FOL-satisfiable iff ...

    $$\begin{array}{ll} \exists x R(x) \wedge \exists x \neg R(x) & \exists x R(x) \wedge \neg\forall x R(x) \\ \dots & \end{array}$$

    - A is FOL-satisfiable iff $\neg A$ is FOL-invalid

The logic landscape

- Classical Propositional Logic (CPL)
  - validity problem decidable
  - ‘zero-order logic’
- First-Order Logic (FOL)
  - validity problem semi-decidable:
    - if A is valid then the decision procedure will answer “yes”;
    - if A is invalid then the decision procedure will either answer “no”, *or loop*.
- Second-Order Logic (SOL), Higher-Order Logics (HOL)
  - undecidable
- ‘in between’: modal logics
  - infinitely many logics
  - ‘many of them’ are decidable (“surprisingly often”)
  - . . . but many of them are not: there are quite simple modal logics that are semi-decidable or even undecidable!

The logic landscape

- Classical Propositional Logic (CPL)
    - validity problem decidable
    - ‘zero-order logic’
- First-Order Logic (FOL)
    - validity problem semi-decidable:
        - if A is valid then the decision procedure will answer “yes”;
        - if A is invalid then the decision procedure will either answer “no”, *or loop*.
- Second-Order Logic (SOL), Higher-Order Logics (HOL)
    - undecidable
- ‘in between’: modal logic**s**
    - infinitely many logics
    - ‘many of them’ are decidable (“surprisingly often”)
    - . . . but many of them are not: there are quite simple modal logics that are semi-decidable or even undecidable!

Modal logics everywhere

- philosophical logic
  - analysis of concepts: necessity and possibility; actions; knowledge; belief; desires, goals and intentions; obligation and permission; qualitative probability, . . .
- artificial intelligence and multiagent systems
  - belief-desire-intention (BDI) agents, normative systems
- theoretical computer science
  - proving properties of (possibly distributed) programs
- semantic web
  - languages for relational structures (ontologies)
- . . .
- mathematical logic
  - logical systems between Classical Propositional Logic and First-Order Logic

$\Longrightarrow$ formal reasoning in modal logics?

The zoo of reasoning methods

“Is formula A valid?”

- Classical Propositional Logic (CPL):
  - Hilbert-style axiomatics; natural deduction
  - Gentzen sequent systems; tableau method
  - resolution
  - *heuristic search (many SAT provers)*
- First-Order Logic (FOL):
  - ...
  - *resolution provers: OTTER, SPASS,...*
- Higher-Order Logic (HOL):
  - *Proof assistants (HOL, Isabelle, Coq,...)*

Reasoning methods for modal logics

- Hilbert-style axiomatics: [Lewis&Langford 32], natural deduction [Prawitz 65]
  - require creativity $\Longrightarrow$ cannot be mechanized
- Gentzen sequent systems: [Došen 85, Wansing 98, Braüner 00, Negri 05, Brünnler 06]
  - 'decorate' proofs by labels $\Longrightarrow$ gets close to semantics
  - difficult to design for some logics (modal logic S5, etc.)
- Resolution: [Enjalbert&Fariñas 89]
  - problem: no simple normal forms in modal logics
- Translation to FOL and resolution: [Ohlbach 88, Fariñas&Herzig 88, Auffray&Enjalbert 89]; MSPASS prover
  - problem: FOL is semi-decidable $\Longrightarrow$ you have to prove that the translation codomain is a decidable fragment of FOL
- Methods integrating SAT provers for CPL with tableaux: [Giunchiglia&Sebastiani 98]; K-SAT theorem prover
- *Tableau methods [Fitting 83]*

Reasoning methods for modal logics

- Hilbert-style axiomatics: [Lewis&Langford 32], natural deduction [Prawitz 65]
  - require creativity $\Longrightarrow$ cannot be mechanized
- Gentzen sequent systems: [Došen 85, Wansing 98, Braüner 00, Negri 05, Brünnler 06]
  - ‘decorate’ proofs by labels $\Longrightarrow$ gets close to semantics
  - difficult to design for some logics (modal logic S5, etc.)
- Resolution: [Enjalbert&Fariñas 89]
  - problem: no simple normal forms in modal logics
- Translation to FOL and resolution: [Ohlbach 88, Fariñas&Herzig 88, Auffray&Enjalbert 89]; MSPASS prover
  - problem: FOL is semi-decidable $\Longrightarrow$ you have to prove that the translation codomain is a decidable fragment of FOL
- Methods integrating SAT provers for CPL with tableaux: [Giunchiglia&Sebastiani 98]; K-SAT theorem prover
- *Tableau methods [Fitting 83]*

Tableau methods for modal logics

“Is there a model for formula A?”

- Equivalent to validity checking:
  - If there is a model for $\neg A$ then A is invalid.
  - If there is no model for $\neg A$ then A is valid.
- Most general method: can be designed for ‘almost all’ modal logics
- Most successful method: tableau provers often match the complexity bounds
- Basic idea of the method: try to build a model by applying the truth conditions

  $\Longrightarrow$ close to semantics

Tableau methods for modal logics

“Is there a model for formula A?”

- Equivalent to validity checking:
  - If there is a model for $\neg A$ then A is invalid.
  - If there is no model for $\neg A$ then A is valid.
- Most general method: can be designed for ‘almost all’ modal logics
- Most successful method: tableau provers often match the complexity bounds
- Basic idea of the method: try to build a model by applying the truth conditions

  $\Longrightarrow$ close to semantics

The idea of this course

Introduce the most important modal logics

. . . via the tableau method

. . . step-by-step

. . . using an implemented tableau prover: LoTREC

Related courses at ESSLLI 2010

- Lutz Strassburger: Introduction to Proof Theory (introductory, 1st week)
- Hans van Ditmarsch: Dynamic epistemic logic (introductory, 2nd week)
- Jan Broersen and Leon van der Torre: Ten problems of deontic logic and normative reasoning in computer science (foundational, 1st week)
- Johan van Benthem and Eric Pacuit: Logic, Rationality, and Intelligent Interaction (workshop, 2nd week)

About the course title

Tarski's World: introduction to FOL

- Alfred Tarski
- examples = scenarios from geometry
- resources:
  - book [Barwise&Etchemendy 91, 93, Barker-Plummer, B&E 04]
  - program (CD)

Kripke's World*s*: introduction to modal logics

- Saul Kripke
- examples = modal logics
- resources:
  - this course
  - book (to come)
  - program: LoTREC

    `http://www.irit.fr/Lotrec`
    online execution and download

Early history: les tableaux de Monsieur Toulouse-LauTREC

Outline of course

Part 1: Modelling with graphs
Part 2: Talking about models
Part 3: The model construction method: basics
Part 4: Logics with simple constraints on models
Part 5: Logics with potential cycles
Part 6: Model checking in LoTREC
Part 7: Logics with transitive closure

Part 1: Modelling with graphs

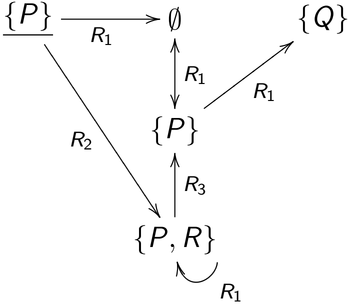
1. Kripke models as graphs

2. Classes of models

Kripke Model [Kripke 59]

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

- $M = (W, R, V)$
  - W: nonempty set (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



Kripke models: terminologies

- Possible worlds = graph nodes

  objects, states
- Valuation = node labeling

  interpretation
- Accessibility relation = edge labeling

  transitions
- Kripke model = labeled graph

  relational model, transition system

What is R?

- Relation between objects:
  wRu iff w is related to u
- 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 occurrence of
  the event I / execution of the program I in w
- . . .

Readings of $R \Longrightarrow$ Properties of R

What is R?

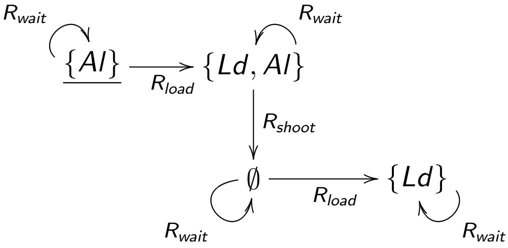
- Relation between objects:
  wRu iff w is related to u
- Alethic:
  wRu iff u is possible given the actual world w
- Temporal:
  wRu iff u is in the future of w
- Epistemic:
  $wR_I u$ iff u is possible for agent I at actual world w
- Deontic:
  wRu iff u is an ideal counterpart of the actual world w
- Dynamic:
  $wR_I u$ iff u is a possible result of the occurrence of
  the event I / execution of the program I in w
- ...

Readings of $R \Longrightarrow$ Properties of R

Actions: the Yale Shooting Problem

A famous scenario [Hanks&McDermott 87]:

- A turkey is initially alive (Al) and a gun is initially unloaded ($\neg Ld$).
- The actions are 'loading the gun', 'waiting for a moment', 'shooting the gun at the turkey' (which is expected to kill the turkey).



Observe: R_{wait} reflexive (= 'skip' program)

Knowledge: muddy children (1)

A famous puzzle:

1. two children come back from the garden, both with mud on their forehead; their father looks at them and says:
   "at least one of you has mud on his forehead"
   then he asks:
   "those who know whether they are dirty, step forward!"
2. nobody steps forward
3. the father asks again:
   "those who know whether they are dirty, step forward!"
4. both simultaneously answer: *"I know!"*

$\Longrightarrow$ model the situation before the father's command

Knowledge: muddy children (2)

- Node labels = propositional variables:

  Md_1 = "child 1 is muddy", etc.
- Edge labels = accessibility relations:

  uR_2v = "child 2 cannot distinguish u and v", etc.

The set of possible worlds and the accessibility relation in the initial situation: . . .

Knowledge: muddy children (2)

- Node labels = propositional variables:

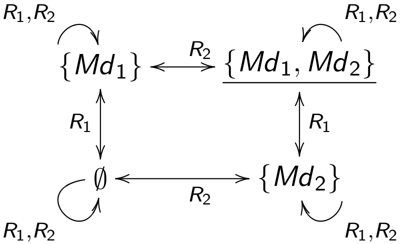
  $Md_1 =$ "child 1 is muddy", etc.
- Edge labels = accessibility relations:

  $uR_2v =$ "child 2 cannot distinguish u and v", etc.

The set of possible worlds and the accessibility relation in the initial situation: ...

Knowledge: muddy children (3)

The set of possible worlds and the accessibility relation in the initial situation (before the father announces $Md_1 \vee Md_2$):

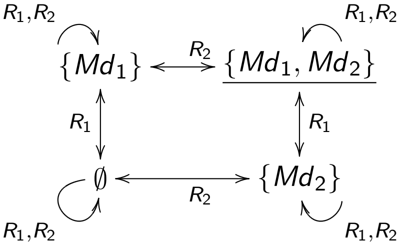


Observe: R_1 and R_2 are equivalence relations: reflexive, transitive and symmetric

The situation after the father announced $Md_1 \vee Md_2$: …

Knowledge: muddy children (3)

The set of possible worlds and the accessibility relation in the initial situation (before the father announces $Md_1 \vee Md_2$):

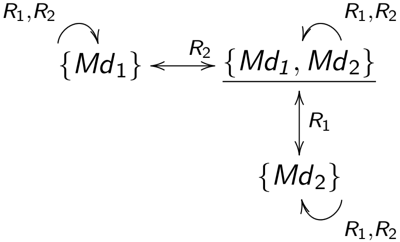


Observe: R_1 and R_2 are equivalence relations: reflexive, transitive and symmetric

The situation after the father announced $Md_1 \vee Md_2$: ...

Knowledge: muddy children (4)

The set of possible worlds and the accessibility relation after the father announced $Md_1 \vee Md_2$:

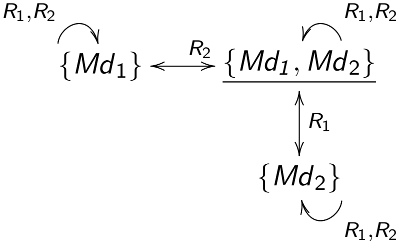


The situation after the first round (when none of the children stepped forward)…

N.B.: can be generalized to an arbitrary number $n \geq 2$ of children

Knowledge: muddy children (4)

The set of possible worlds and the accessibility relation after the father announced $Md_1 \vee Md_2$:



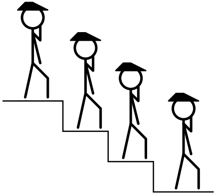
The situation after the first round (when none of the children stepped forward)...

N.B.: can be generalized to an arbitrary number $n \geq 2$ of children

Knowledge: and now for something different

Example (thanks T. de Lima)

There are n stairs and n agents 1,. . . , n.
On every step k stands agent k.
Each agent wears a hat that is either black or white.
Agent k can see the hats of the agents $l > k$.
Agent k cannot see the hats of the agents $\leq k$.

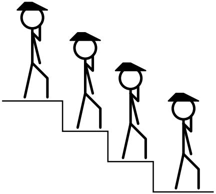


Agents are only able to announce "black" or "white".
Find a sequence of n announcements such that at least $n - 1$ agents correctly announce the colour of their own hat (they can discuss beforehand).

Knowledge: and now for something different

Example (thanks T. de Lima)

There are n stairs and n agents 1,..., n.
On every step k stands agent k.
Each agent wears a hat that is either black or white.
Agent k can see the hats of the agents $l > k$.
Agent k cannot see the hats of the agents $\leq k$.



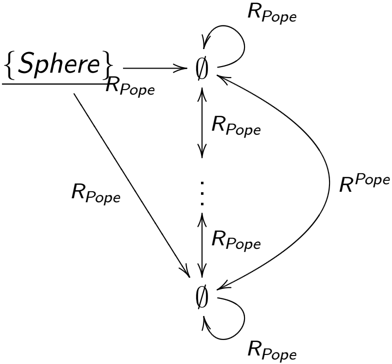
Agents are only able to announce "black" or "white".
Find a sequence of n announcements such that at least $n-1$ agents correctly announce the colour of their own hat (they can discuss beforehand).

Belief: the earth is flat

A historical example [1630]: "The Pope believes the earth is flat."

- Node labels: *Sphere* = "Earth is a sphere"
- Edge labels:

  $uR_{Pope}v$ = "at u, v is compatible with Pope's beliefs"



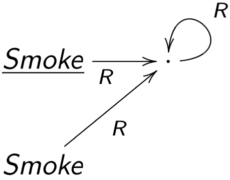
Observe: R_{Pope} is not reflexive; is transitive and *Euclidean* (for all u, v_1, v_2, if uRv_1 and uRv_2 then v_1Rv_2)

Obligations

An actual example (since 2008 in France):
"It is forbidden to smoke in restaurants."

- Node labels: *Smoke* = "somebody is smoking"
- Edge labels:

uRv = "v is a world where everything obligatory in u is true",
"v is an ideal world w.r.t. u",
"at u, v is a permitted state"



Observe: R is not reflexive; should not necessarily be transitive; is *serial* (for all u there is v such that uRv)

Knowledge and obligations: the norm-violating muddy children

Exercise

Add to the muddy children scenario:
"none of the children should be muddy"

In your Kripke model, do the children know that it is obligatory to be clean? If so, find a Kripke model where they do not know that obligation.

In your new Kripke model, does child 1 know that child 2 does not know the obligation? If so, . . .

Knowledge and obligations: the norm-violating muddy children

Exercise

Add to the muddy children scenario:
“none of the children should be muddy”

In your Kripke model, do the children know that it is obligatory to be clean? If so, find a Kripke model where they do not know that obligation.

In your new Kripke model, does child 1 know that child 2 does not know the obligation? If so, . . .

Knowledge and obligations: the norm-violating muddy children

Exercise

Add to the muddy children scenario:
“none of the children should be muddy”

In your Kripke model, do the children know that it is obligatory to be clean? If so, find a Kripke model where they do not know that obligation.

In your new Kripke model, does child 1 know that child 2 does not know the obligation? If so, . . .

Building a graph in LoTREC

1. `http://www.irit.fr/Lotrec` (Capital "L")
    - Webstart
    - or, *Download* ⟹ *Executable* to get *LoTREC_2.0.zip*
        - unzip, then run file *run.bat*
2. Open a new logic (menu 'Logic')
3. Add a new rule ('Rules' tab):
    - no conditions
    - in the action part:

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

(capital first letter ⟹ constant, small first letter ⟹ variable)

4. Edit the default strategy ('Strategies' tab):
    - call the new rule (double click)
5. Click on "Build Premodels"

Part 1: Modelling with graphs

1. Kripke models as graphs

2. Classes of models

Classes of models

- A class of models can be defined by
    - constraints on the accessibility relation
    - constraints on the valuation
- Constraints depend on the concepts we want to model
    - time
    - events, programs
    - actions
    - knowledge
    - belief
    - obligations
    - ...
- Mathematical properties?
    - satisfiability in class decidable?
    - complexity?

Constraints on a single relation R

- Transitive
- Reflexive
- Serial: for all u exists v s.th. uRv
- Deterministic: for all u, v_1, v_2, if uRv_1 and uRv_2 then $v_1{=}v_2$
- Euclidean: for all u, v_1, v_2, if uRv_1 and uRv_2 then v_1Rv_2
- Linear: for all u, v_1, v_2, if uRv_1 and uRv_2 then v_1Rv_2 or v_2Rv_1
- Symmetric
- Equivalence relation: reflexive, transitive, symmetric
- Confluent (Church-Rosser)
- no infinite R-chain (conversely well-founded)
- Universal: $R = W \times W$
- Singleton models: $\{M : card(W) = 1\}$
- ...

Constraints involving several relations

- Inclusion: $R_I \subseteq R_J$
- Union: $R_I = R_J \cup R_K$
- Converse: $R_J = (R_I)^{-1}$
- Reflexive and transitive closure: $R_J = (R_I)^*$
- Permutation: $R_I \circ R_J \subseteq R_J \circ R_I$
- Confluence: ...
- ...

Constraints on the valuation V

- names for worlds:
  if $N \in V(w)$ and $N \in V(u)$ then $w = u$
    - ‘nominals’ (hybrid logic)
    - ‘objects’ (description logics)

- R is persistent (alias hereditary):
  if $P \in V(w)$ and wRu then $P \in V(u)$
    - intuitionistic implication

Time: constraints

- ‘the (non-strict) future includes the present’
  $\Longrightarrow$ R reflexive
- ‘strict future excludes the present’
  $\Longrightarrow$ R irreflexive
- ‘future of future is future’
  $\Longrightarrow$ R transitive
- ‘there is always a future state’
  $\Longrightarrow$ R serial
- ‘time is linear’
  $\Longrightarrow$ R linear
- ‘time will come to an end’
  $\Longrightarrow$ no infinite R-chain (conversely well-founded)
- $R_I =$ ‘future’, $R_J =$ ‘past’ (or: $R_I =$ ‘tomorrow’, $R_J =$ ‘yesterday’)
  - $R_J = (R_I)^{-1}$
- $R_I =$ ‘future’, $R_J =$ ‘next’
  - $R_I = (R_J)^*$

Knowledge: constraints

- $R =$ ‘indistinguishability’
  $\Longrightarrow$ Equivalence relation:
  - reflexive (‘knowledge is true’),
    transitive (‘I know what I know’),
    Euclidean (‘I know what I don’t know’)
  - same as: reflexive, transitive, and symmetric
- Confluence instead of Euclideanity [Lenzen]

Belief: constraints

- actual world not necessarily in the worlds compatible with the agent's belief
  $\Longrightarrow$ R not necessarily reflexive
- transitive, Euclidean, serial

Programs and events: constraints

- $I_3 =$ 'nondeterministic composition of programs I_1 and I_2'
  $\Longrightarrow R_{I_3} = R_{I_1} \cup R_{I_2}$
- $I_2 =$ 'execution of program I_1 the other way round'
  $\Longrightarrow R_{I_2} = (R_{I_1})^{-1}$
- $I_2 =$ 'iteration of program I_1'
  $\Longrightarrow R_{I_2} = (R_{I_1})^{*}$

Knowledge and events: constraints

R_I = effect of event I
R_J = indistinguishable for agent J

- 'no forgetting' (alias 'perfect recall')
  $\Longrightarrow R_I \circ R_J \subseteq R_J \circ R_I$

$$\begin{tikzcd}
\cdot \arrow[r, dashed, "R_I"] & \cdot \\
\cdot \arrow[u, dashed, "R_J"] \arrow[r, "R_I"'] & \cdot \arrow[u, "R_J"']
\end{tikzcd}$$

- 'no learning' (alias 'no miracles')
  $\Longrightarrow R_J \circ R_I \subseteq R_I \circ R_J$

Closing under constraints in LoTREC

- Add a rule (‘Rules’ tab) which closes under reflexivity:
  condition: `isNewNode w`
  action: `link w w R`
  (capital first letter $\Longrightarrow$ constant, small first letter $\Longrightarrow$ variable)
- Exercise: close `R` under transitivity
- Exercise: make `R` persistent

Outline of course

Part 1: Modelling with graphs
Part 2: Talking about models
Part 3: The model construction method: basics
Part 4: Logics with simple constraints on models
Part 5: Logics with potential cycles
Part 6: Model checking in LoTREC
Part 7: Logics with transitive closure

Part 2: Talking about models

3 The modal language

4 Truth conditions

5 Reasoning in modal logics

6 The standard translation

Talking about graphs in the first-order language

- In the language of First-Order Logic FOL:
  - nodes = variables
  - unary predicates for the node labels
  - binary predicates for the edge labels
  - quantify over nodes

Examples

$\exists w(Md_1(w) \wedge Md_2(w))$

$\exists w(\neg Md_1(w) \wedge \neg Md_2(w))$

$\forall w(\exists u(R_1(w,u) \wedge Md_1(u)) \wedge \exists u(R_1(w,u) \wedge \neg Md_1(u)))$

$\forall w(Md_1(w) \rightarrow \neg Md_2(w))$ (after father's announcement)

$\forall w(R_{load}(w,u) \rightarrow Ld(u))$ $\forall w((Ld(w) \wedge R_{shoot}(w,u)) \rightarrow \neg Al(u))$

Talking about graphs in a modal language

- Don't mention nodes, only talk about their properties
- View the graph locally (sitting at a node)
  - "P" means "P labels the actual node"
  - $\Box_l P$ means "P labels *every* node accessible from the actual node via an edge labeled l"

    $\Box_{load} Ld$ (Yale Shooting Problem)

    $\Box_1 Md_2$ (Muddy Children Puzzle)
  - $\Diamond_l P$ means "P labels *some* node accessible from the actual node via an edge labeled l"

    $\Diamond_{load} Ld$

    $\Diamond_1 Md_1$
- use Boolean operators

  $\neg\Diamond_1 Md_2$ (in the actual world)

  $\Diamond_1 Md_1 \wedge \Diamond_1 \neg Md_1$

  $\Box_1(Md_1 \vee Md_2)$ (after the father has announced $Md_1 \vee Md_2$)

  $\Box_l P \rightarrow \Diamond_l P$

Talking about graphs in a modal language

- Don't mention nodes, only talk about their properties
- View the graph locally (sitting at a node)
  - "P" means "P labels the actual node"
  - $\Box_l P$ means "P labels *every* node accessible from the actual node via an edge labeled l"

    $\Box_{load} Ld$ (Yale Shooting Problem)

    $\Box_1 Md_2$ (Muddy Children Puzzle)
  - $\Diamond_l P$ means "P labels *some* node accessible from the actual node via an edge labeled l"

    $\Diamond_{load} Ld$

    $\Diamond_1 Md_1$
- use Boolean operators

  $\neg\Diamond_1 Md_2$ (in the actual world)

  $\Diamond_1 Md_1 \wedge \Diamond_1 \neg Md_1$

  $\Box_1(Md_1 \vee Md_2)$ (after the father has announced $Md_1 \vee Md_2$)

  $\Box_l P \rightarrow \Diamond_l P$

Reading the modal operators: necessity and possibility

- Monomodal (just one operator)

  $$\Diamond A \quad = \quad \mathrm{M}A \quad = \quad \text{“}A\text{ is possible”}$$
  $$\Box A \quad = \quad \mathrm{L}A \quad = \quad \text{“}A\text{ is necessary”}$$

- Multimodal version:

  $$\Diamond_I A \quad = \quad \langle I \rangle A \quad = \quad \text{“}A\text{ is possible w.r.t. }I\text{”}$$
  $$\Box_I A \quad = \quad [I]A \quad = \quad \ldots$$

Necessity and possibility: two different usages

1. Logical/metaphysical/. . . necessity and possibility
   $\Longrightarrow$ modal logic in the narrow sense

2. Any expression that is used to qualify the truth of a judgement
   Example: "it is raining":
   - "it will always rain" (temporal)
   - "it will sometimes rain" (temporal)
   - "it will rain tomorrow" (temporal)
   - "it is known that it is raining" (epistemic)
   - "it is believed that it is raining" (doxastic)
   - "it will rain after sunset" (dynamic)
   - "it should be the case that it is raining" (deontic)
   - "it is permitted that it is raining" (deontic)
   - . . .

   Common feature: not truth-functional
   - no function f s.th. $truthvalue(\Diamond A) = f(truthvalue(A))$

   $\Longrightarrow$ *modal logics in the large sense*

Temporal operators

$\mathrm{F}A$ = “*A* will be true at *some* time point in the future”

= “*A* will eventually be true”

$\mathrm{G}A$ = “*A* will be true at *every* time point in the future”

= “*A* will be true henceforth”

$\mathrm{P}A$ = “*A* was true at *some* time point in the past”

$\mathrm{H}A$ = “*A* was true at *every* time point in the past”

$A\mathrm{U}B$ = “*A* until *B*”

$A\mathrm{B}B$ = “*A* before *B*”

$A\mathrm{S}B$ = “*A* since *B*”

$\mathrm{X}A$ = “*A* will be true at the *next* time point”

N.B.: $\mathrm{G}A$ should imply $\mathrm{F}A$, and $\mathrm{H}A$ should imply $\mathrm{P}A$

Dynamic operators

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

N.B.: programs may be nondeterministic; $\mathrm{Feasible}_I A$ does not imply $\mathrm{After}_I A$

Epistemic and doxastic operators

- *episteme* $= \epsilon\pi\iota\sigma\tau\eta\mu\eta =$ 'know' (Greek)
- *doxa* $= \delta o\xi\alpha =$ 'believe' (Greek)

$$\begin{aligned}
\mathrm{Bel}_I A &= \text{“agent } I \text{ believes that } A\text{”} \\
&= \text{“} A \text{ is true in all possible worlds compatible} \\
& \quad \text{with what } I \text{ believes”} \\
\mathrm{K}_I A &= \text{“agent } I \text{ knows that } A\text{”} \\
&= \text{“} A \text{ is true in all possible worlds compatible} \\
& \quad \text{with what } I \text{ knows”} \\
\hat{\mathrm{Bel}}_I A &= \text{“} A \text{ is compatible with } I\text{'s beliefs”} \\
\hat{\mathrm{K}}_I A &= \text{“} A \text{ is compatible with } I\text{'s knowledge”}
\end{aligned}$$

N.B.: $\mathrm{K}_I A$ should imply $\mathrm{Bel}_I A$

Deontic operators

- $\delta\epsilon o\nu$ = "binding" (Greek)

$$
\begin{array}{lcl}
\mathrm{O}A & = & \text{“}A\text{ is obligatory”} \\
\mathrm{P}A & = & \text{“}A\text{ is permitted”} \\
\ldots & = & \text{“}A\text{ is forbidden”} = \neg \mathrm{P}A \\
\ldots & = & \text{“}A\text{ is omissible”} = \neg \mathrm{O}A
\end{array}
$$

Can be relative to a normative system:
$\mathrm{O}_{France} \neg Smoke$, but $\neg \mathrm{O}_{Portugal} \neg Smoke$

Can be relative to an agent (personal obligation)

N.B.: $\mathrm{O}A$ should imply $\mathrm{P}A$

Mixing several kinds of operators

- Epistemic and event operators [Plaza, Baltag&Moss, Gerbrandy, van Ditmarsch, van der Hoek&Kooi,...]:
  $\neg \mathrm{K}_1 \mathit{Md_1} \wedge \mathrm{After}_{\mathit{Md_1}!} \mathrm{K}_1 \mathit{Md_1}$
  (event = announcement of $\mathit{Md_1}$)
  $\neg \mathrm{K}_1 \mathit{Md_1} \wedge \mathrm{After}_{\neg \mathrm{K}_2 \mathit{Md_2}!} \mathrm{K}_1 \mathit{Md_1}$
- Epistemic and temporal operators [Fagin, Halpern, Moses, Vardi]:

  ...
- Doxastic and temporal operators:

  ...
- Epistemic and deontic operators:
  $\mathrm{O}_{\mathit{France}} \neg \mathit{Smoke} \wedge \neg \mathrm{K}_{\mathit{I}} \mathrm{O}_{\mathit{France}} \neg \mathit{Smoke}$
- ...

  $\Longrightarrow$ Multi-dimensional modal logics

Mixing several kinds of operators

- Epistemic and event operators [Plaza, Baltag&Moss, Gerbrandy, van Ditmarsch, van der Hoek&Kooi,...]:
  $\neg \mathrm{K}_1 \mathit{Md_1} \wedge \mathrm{After}_{\mathit{Md_1}!} \mathrm{K}_1 \mathit{Md_1}$
  (event = announcement of $\mathit{Md_1}$)
  $\neg \mathrm{K}_1 \mathit{Md_1} \wedge \mathrm{After}_{\neg \mathrm{K}_2 \mathit{Md_2}!} \mathrm{K}_1 \mathit{Md_1}$
- Epistemic and temporal operators [Fagin, Halpern, Moses, Vardi]:

  ...
- Doxastic and temporal operators:

  ...
- Epistemic and deontic operators:
  $\mathrm{O}_{\mathit{France}} \neg \mathit{Smoke} \wedge \neg \mathrm{K}_{\mathit{I}} \mathrm{O}_{\mathit{France}} \neg \mathit{Smoke}$
- ...
  $\Longrightarrow$ Multi-dimensional modal logics

Implications, implications . . .

- Strict implication [Lewis&Langford]
  $A \prec B \quad = \quad$ "A strictly implies B"
  - like $\rightarrow$, but $A \prec (B \prec A)$ invalid
  - $= \Box(A \rightarrow B)$
- Intuitionistic implication [Brouwer, Gödel, Kripke]
  $A \Rightarrow B \quad = \quad$ "A intuitionistically implies B"
  - like $\rightarrow$, but excluded middle $A \vee (A \Rightarrow \bot)$, . . . invalid
- Conditional [Lewis]
  $A \Box\!\!\Rightarrow B \quad = \quad$ "if A then B"
  - like $\Rightarrow$, but strengthening in the antecedent $(A \Box\!\!\Rightarrow B) \rightarrow ((A \wedge C) \Box\!\!\Rightarrow B)$ invalid
- Conditional obligation [Chellas,. . .]
  $\mathrm{O}(A|B) \quad = \quad$ "it ought to be that if A, then B"
  - $\mathrm{O}(A|B)$ different from $\mathrm{O}(A \rightarrow B)$

Duality

- Intuitively:

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

- Abstracting:

  $$\begin{array}{lcl} \Diamond_I A & \leftrightarrow & \neg \Box_I \neg A \\ \Box_I A & \leftrightarrow & \neg \Diamond_I \neg A \end{array}$$

- Options for the choice of the primitives:
  - take both $\Diamond_I$ and $\Box_I$ as primitive
  - take $\Diamond_I$ as primitive, and set $\Box_I A \stackrel{\mathrm{def}}{=} \neg \Diamond_I \neg A$
  - take $\Box_I$ as primitive, and set $\Diamond_I A \stackrel{\mathrm{def}}{=} \neg \Box_I \neg A$

Duality

- Intuitively:

  $\hat{\mathrm{K}}_I A \quad \leftrightarrow \quad \neg \mathrm{K}_I \neg A$
  $\mathrm{P}_I A \quad \leftrightarrow \quad \neg \mathrm{O}_I \neg A$
  $\mathrm{F} A \quad \leftrightarrow \quad \neg \mathrm{G} \neg A$
  $\mathrm{After}_I A \quad \leftrightarrow \quad \neg \mathrm{Feasible}_I \neg A$
  $\ldots$
- Abstracting:

  $\Diamond_I A \quad \leftrightarrow \quad \neg \Box_I \neg A$
  $\Box_I A \quad \leftrightarrow \quad \neg \Diamond_I \neg A$
- Options for the choice of the primitives:
  - take both $\Diamond_I$ and $\Box_I$ as primitive
  - take $\Diamond_I$ as primitive, and set $\Box_I A \stackrel{\mathrm{def}}{=} \neg \Diamond_I \neg A$
  - take $\Box_I$ as primitive, and set $\Diamond_I A \stackrel{\mathrm{def}}{=} \neg \Box_I \neg A$

How define a language?

- Set of node labels $\mathcal{P} = \{P, Q, \ldots\}$ ('propositional variables')
- Set of edge labels $\mathcal{I} = \{I, J, \ldots\}$ ('indexes')
- Language $=$ set of well-formed formulas
- Language is defined by BNF:

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

  where P ranges over $\mathcal{P}$ and I ranges over $\mathcal{I}$
  (unary modal operators only)
- Convention: when $\mathcal{I} = \{I\}$ then
  - write $\Box A$ instead of $[I]A$
  - write $\Diamond A$ instead of $\langle I \rangle A$

How define a language in LoTREC?

- Prenex form: a LoTREC formula is
  - a propositional variable P, or
  - an expression of the form $op(Arg_1, \ldots, Arg_n)$ where
    - op is the name of a logical operator
    - the Arg_i are either formulas or in the index set $\mathcal{I}$

$$\begin{aligned} \neg A &= not(A) \\ A \wedge B &= and(A, B) \\ A \vee B &= or(A, B) \\ &\ldots \end{aligned} \qquad \begin{aligned} \mathrm{Bel}_I A &= Bel(I, A) \\ \mathrm{K}_I A &= Knows(I, A) \\ \hat{\mathrm{K}}_I A &= Poss(I, A) \\ &\ldots \\ A \mathrm{U} B &= Until(A, B) \\ &\ldots \\ A \Rightarrow B &= ifThen(A, B) \\ &\ldots \end{aligned}$$

N.B.: may write $op\ Arg_1\ \ldots\ Arg_n$ (parentheses not needed)

- Adding a new connector:

  'Connectors' tab $\Longrightarrow$ name, arity, display mode

Part 2: Talking about models

3 The modal language

4 Truth conditions

5 Reasoning in modal logics

6 The standard translation

Truth conditions

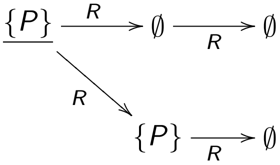
Evaluate a formula A in a *pointed model* (M, w),
where $M = (W, R, V)$ and $w \in W$ ('the actual world')

- Atoms
  - $M, w \Vdash P$ iff $P \in V(w)$
- Boolean operators
  - $M, w \Vdash \neg A$ iff $M, w \nVdash A$
  - $M, w \Vdash A \wedge B$ 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

Example

For the pointed model (M, w)



(actual world underlined) we have:
$M, w \Vdash P$
$M, w \Vdash \Diamond\neg P$
$M, w \Vdash \Box\Diamond\neg P$
But: $M, w \nVdash \Box P$.

Truth conditions

- Multi-modal operators
  - $M, w \Vdash \langle I \rangle A$ iff there exists u s.th. $wR_I u$ and $M, u \Vdash A$
  - $M, w \Vdash [I]A$ iff ...

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

Truth conditions

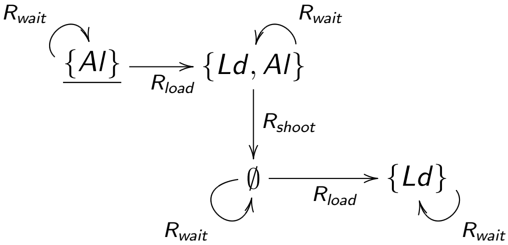
- Temporal operators (linear time)
  - $M, w \Vdash \mathrm{X}A$ iff there exists u s.th. wRu and $M, u \Vdash A$
  - $M, w \Vdash \mathrm{F}A$ iff there exists n,u s.th. $wR^n u$ and $M, u \Vdash A$
  - $M, w \Vdash A\mathrm{U}B$ iff there exists u s.th. $wR^* u$ and
    - $M, u \Vdash B$
    - $M, v \Vdash A$ for all v s.th. ($wR^* v$ and $vR^+ u$)

$$\begin{tikzcd}
\underset{w}{A} \arrow[r] & A \arrow[r] & \cdots \arrow[r] & A \arrow[r] & \underset{u}{B}
\end{tikzcd}$$

  - ...
- Implications
  - $M, w \Vdash A \Rightarrow B$ iff for all u, if wRu then $M, u \nVdash A$ or $M, u \Vdash B$
    (both for strict and intuitionistic implication;
    difference: preservation condition)
  - $M, w \Vdash A \mathrel{\Box\!\!\Rightarrow} B$ iff for all u,
    if $u \in min_R(\{v \mid wRv \text{ and } M, v \Vdash A\})$ then $M, u \Vdash B$
    (conditional)

Talking about actions

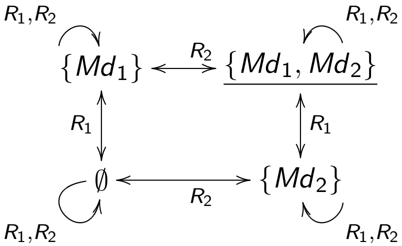
The Yale Shooting Problem:



$M, w \Vdash \mathrm{After}_{load}\mathrm{After}_{wait}\mathrm{After}_{shoot}\neg Al$
$M, w \Vdash \mathrm{After}_{shoot}\bot$
$M, w \Vdash \mathrm{After}_{load}\neg\mathrm{After}_{shoot}\bot$

Talking about knowledge (1)

Muddy children puzzle, initial situation:



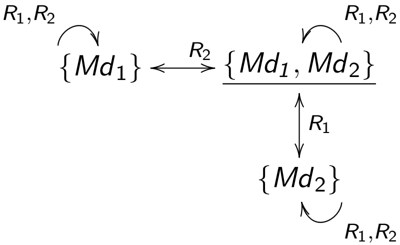
$M, w \Vdash \mathrm{K}_1 Md_2$

$M, w \Vdash \hat{\mathrm{K}}_1 Md_1 \wedge \hat{\mathrm{K}}_1 \neg Md_1$

$M, w \Vdash \hat{\mathrm{K}}_1 \hat{\mathrm{K}}_2 (\neg Md_1 \wedge \neg Md_2)$

Talking about knowledge (2)

Muddy children puzzle, after father has announced $Md_1 \vee Md_2$:



$M, w \nVdash \hat{\mathrm{K}}_1\hat{\mathrm{K}}_2(\neg Md_1 \wedge \neg Md_2)$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2(Md_1 \vee Md_2)$
$M, w \nVdash \mathrm{K}_1 Md_1$

Talking about knowledge (3)

Muddy children puzzle, after the first round (when none of the children stepped forward):

...

$M, w \Vdash \mathrm{K}_1(Md_1 \wedge Md_2) \wedge \mathrm{K}_2(Md_1 \wedge Md_2)$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2) \wedge \mathrm{K}_2\mathrm{K}_1(Md_1 \wedge Md_2) \wedge$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2\mathrm{K}_1(Md_1 \wedge Md_2) \wedge \mathrm{K}_2\mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2)$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2\mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2) \wedge \ldots$
...
$M, w \Vdash \mathrm{CK}_{\{1,2\}}(Md_1 \wedge Md_2)$ (common knowledge)

Truth condition:

- $M, w \Vdash \mathrm{CK}_{\{1,2\}}A$ iff for all u, if $w(R_1 \cup R_2)^* u$ then $M, u \Vdash A$

Talking about knowledge (3)

Muddy children puzzle, after the first round (when none of the children stepped forward):

. . .

$M, w \Vdash \mathrm{K}_1(Md_1 \wedge Md_2) \wedge \mathrm{K}_2(Md_1 \wedge Md_2)$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2) \wedge \mathrm{K}_2\mathrm{K}_1(Md_1 \wedge Md_2) \wedge$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2\mathrm{K}_1(Md_1 \wedge Md_2) \wedge \mathrm{K}_2\mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2)$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2\mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2) \wedge \ldots$
. . .
$M, w \Vdash \mathrm{CK}_{\{1,2\}}(Md_1 \wedge Md_2)$ (common knowledge)

Truth condition:

- $M, w \Vdash \mathrm{CK}_{\{1,2\}}A$ iff for all u, if $w(R_1 \cup R_2)^* u$ then $M, u \Vdash A$

Talking about knowledge (3)

Muddy children puzzle, after the first round (when none of the children stepped forward):

…

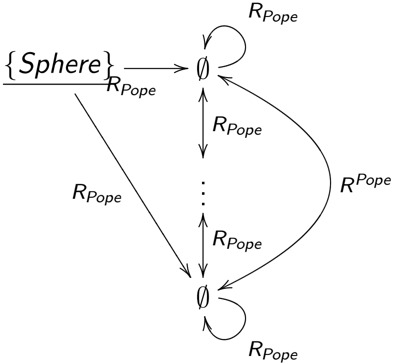
$M, w \Vdash \mathrm{K}_1(Md_1 \wedge Md_2) \wedge \mathrm{K}_2(Md_1 \wedge Md_2)$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2) \wedge \mathrm{K}_2\mathrm{K}_1(Md_1 \wedge Md_2) \wedge$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2\mathrm{K}_1(Md_1 \wedge Md_2) \wedge \mathrm{K}_2\mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2)$
$M, w \Vdash \mathrm{K}_1\mathrm{K}_2\mathrm{K}_1\mathrm{K}_2(Md_1 \wedge Md_2) \wedge \ldots$
…
$M, w \Vdash \mathrm{CK}_{\{1,2\}}(Md_1 \wedge Md_2)$ (common knowledge)

Truth condition:

- $M, w \Vdash \mathrm{CK}_{\{1,2\}}A$ iff for all u, if $w(R_1 \cup R_2)^* u$ then $M, u \Vdash A$

Talking about belief

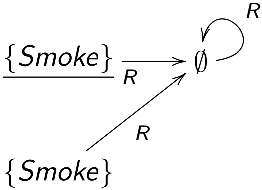
"The Pope believes the earth is flat."



$M, w \Vdash Sphere \wedge \mathrm{Bel}_{Pope} \neg Sphere$

Talking about obligations

"It is forbidden to smoke in restaurants."

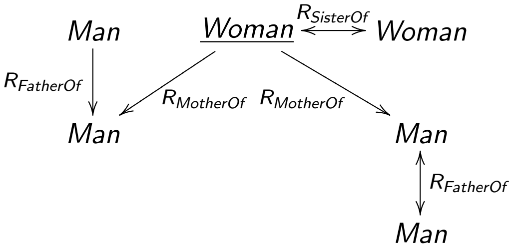


$M, w \Vdash Smoke \wedge \mathrm{O}\neg Smoke \wedge \neg \mathrm{P}\, Smoke$

Talking about objects and their relations

The genealogy ontology:

- Propositional variables: *Man*, *Woman*, … ('concepts')
- Modal operators:
  FatherOf, *MotherOf*, *BrotherOf*, *GrandfatherOf*, … ('roles')



$M, w \Vdash Woman \wedge \exists SisterOf.Woman \wedge \forall MotherOf.Man$
$M, w \Vdash \exists MotherOf.\exists FatherOf.Man$

Part 2: Talking about models

3 The modal language

4 Truth conditions

5 Reasoning in modal logics

6 The standard translation

Model checking

$$\boxed{\text{Given } A\text{, pointed model } (M, w)\text{: do we have } M, w \Vdash A?}$$

Validity in a Kripke model

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

Given formula A, model M: is A valid in M?

Example

Formula $\Box P$ is valid in the model

$$\underline{\emptyset} \xrightarrow{R} \{P\}$$

The formulas $\neg P$ and $\Box P \to P$ are not valid in M.

Validity in a class of Kripke models

- K = the class of all **K**ripke models
- A is valid in the class of models $\mathcal{C} \subseteq$ K iff

  for *all* models M in $\mathcal{C}$: A is valid in M

  notation: $\models_{\mathcal{C}} A$

$$\boxed{\text{Given formula } A\text{, class of models } \mathcal{C}\text{: is } A \text{ valid in } \mathcal{C}?}$$

Examples

$\Diamond P \leftrightarrow \neg\Box\neg P$ is valid in K
$\Box(P \vee \neg P)$ is valid in K
$\Box P \wedge \Box Q \rightarrow \Box(P \wedge Q)$ is K-valid
$\Box P \rightarrow P$ is K-invalid (being invalid in our example model)

Examples of validity in a class of models

- Reflexive models (KT)
  $\Box P \rightarrow P$ valid
  $\Box A \rightarrow A$ valid, for any formula A ($A =$ schematic variable)
- Transitive models (K4)
  $\Diamond\Diamond A \rightarrow \Diamond A$ valid, for any formula A (alias $\Box A \rightarrow \Box\Box A$)
- Reflexive and transitive models: S4
  valid: . . .
- Symmetric relation (KB)
  valid: $A \rightarrow \Box\Diamond A$ (alias $\Diamond\Box A \rightarrow A$)
- Euclidean relation (K5) (alias. . . valid: $\Diamond A \rightarrow \Box\Diamond A$
- Equivalence relation (S5)
  valid: . . .

Examples of validity in a class of models

- Reflexive models (KT)
  $\Box P \rightarrow P$ valid
  $\Box A \rightarrow A$ valid, for any formula A (A = schematic variable)
- Transitive models (K4)
  $\Diamond\Diamond A \rightarrow \Diamond A$ valid, for any formula A (alias $\Box A \rightarrow \Box\Box A$)
- Reflexive and transitive models: S4
  valid: ...
- Symmetric relation (KB)
  valid: $A \rightarrow \Box\Diamond A$ (alias $\Diamond\Box A \rightarrow A$)
- Euclidean relation (K5) (alias... valid: $\Diamond A \rightarrow \Box\Diamond A$
- Equivalence relation (S5)
  valid: ...

Examples of validity in a class of models

- Reflexive models (KT)
  $\Box P \rightarrow P$ valid
  $\Box A \rightarrow A$ valid, for any formula A ($A =$ schematic variable)
- Transitive models (K4)
  $\Diamond\Diamond A \rightarrow \Diamond A$ valid, for any formula A (alias $\Box A \rightarrow \Box\Box A$)
- Reflexive and transitive models: S4
  valid: . . .
- Symmetric relation (KB)
  valid: $A \rightarrow \Box\Diamond A$ (alias $\Diamond\Box A \rightarrow A$)
- Euclidean relation (K5) (alias. . . valid: $\Diamond A \rightarrow \Box\Diamond A$
- Equivalence relation (S5)
  valid: . . .

Examples of validity in a class of models (ctd.)

- Serial models: for every w there is w' such that wRw'
  valid: ...
- Deterministic models:
  valid: ...
- Confluence:
  valid: $\Diamond\Box A \rightarrow \Box\Diamond A$
- Linearity:
  valid: $\Diamond A \land \Diamond B \ \rightarrow\ (\Diamond(A \land \Diamond B)) \lor (\Diamond(\Diamond A \land B))$
- Singleton models: $\{M : card(W) = 1\}$
  valid: ...
- Inclusion of R_1 in R_2
  valid: $\Box_2 A \rightarrow \Box_1 A$ (alias $\Diamond_1 A \rightarrow \Diamond_2 A$)
- Permutation: $R_1 \circ R_2 \subseteq R_2 \circ R_1$
  valid: $\Diamond_1\Diamond_2 A \rightarrow \Diamond_2\Diamond_1 A$ (alias $\Box_2\Box_1 A \rightarrow \Box_1\Box_2 A$)
- ...

Examples of validity in a class of models (ctd.)

- Serial models: for every w there is w' such that wRw'
  valid: . . .
- Deterministic models:
  valid: . . .
- Confluence:
  valid: $\Diamond\Box A \to \Box\Diamond A$
- Linearity:
  valid: $\Diamond A \wedge \Diamond B \ \to \ (\Diamond(A \wedge \Diamond B)) \vee (\Diamond(\Diamond A \wedge B))$
- Singleton models: $\{M : card(W) = 1\}$
  valid: . . .
- Inclusion of R_1 in R_2
  valid: $\Box_2 A \to \Box_1 A$ (alias $\Diamond_1 A \to \Diamond_2 A$)
- Permutation: $R_1 \circ R_2 \subseteq R_2 \circ R_1$
  valid: $\Diamond_1\Diamond_2 A \to \Diamond_2\Diamond_1 A$ (alias $\Box_2\Box_1 A \to \Box_1\Box_2 A$)
- . . .

Examples of validity in a class of models (ctd.)

- Serial models: for every w there is w' such that wRw'
  valid: ...
- Deterministic models:
  valid: ...
- Confluence:
  valid: $\Diamond\Box A \to \Box\Diamond A$
- Linearity:
  valid: $\Diamond A \land \Diamond B \;\to\; (\Diamond(A \land \Diamond B)) \lor (\Diamond(\Diamond A \land B))$
- Singleton models: $\{M : card(W) = 1\}$
  valid: ...
- Inclusion of R_1 in R_2
  valid: $\Box_2 A \to \Box_1 A$ (alias $\Diamond_1 A \to \Diamond_2 A$)
- Permutation: $R_1 \circ R_2 \subseteq R_2 \circ R_1$
  valid: $\Diamond_1\Diamond_2 A \to \Diamond_2\Diamond_1 A$ (alias $\Box_2\Box_1 A \to \Box_1\Box_2 A$)
- ...

Examples of validity in a class of models (ctd.)

- Serial models: for every w there is w' such that wRw'
  valid: ...
- Deterministic models:
  valid: ...
- Confluence:
  valid: $\Diamond\Box A \rightarrow \Box\Diamond A$
- Linearity:
  valid: $\Diamond A \wedge \Diamond B \;\rightarrow\; (\Diamond(A \wedge \Diamond B)) \vee (\Diamond(\Diamond A \wedge B))$
- Singleton models: $\{M : card(W) = 1\}$
  valid: ...
- Inclusion of R_1 in R_2
  valid: $\Box_2 A \rightarrow \Box_1 A$ (alias $\Diamond_1 A \rightarrow \Diamond_2 A$)
- Permutation: $R_1 \circ R_2 \subseteq R_2 \circ R_1$
  valid: $\Diamond_1\Diamond_2 A \rightarrow \Diamond_2\Diamond_1 A$ (alias $\Box_2\Box_1 A \rightarrow \Box_1\Box_2 A$)
- ...

Logical consequence in a class of models

- B is a global logical consequence of A in class $\mathcal{C}$ iff *for all* M in $\mathcal{C}$: if A is valid in M then if B is valid in M
- notation: $A \models_{\mathcal{C}} B$

Given formulas A, B, class of models $\mathcal{C}$: do we have $A \models_{\mathcal{C}} B$?

- B is a local logical consequence of A in class $\mathcal{C}$ iff *for all* M in $\mathcal{C}$ and w in M: if $M, w \Vdash A$ then $M, w \Vdash B$
- Difference: $\Box A$ is a global logical consequence of A, but not a local consequence.

Proposition

B is a local logical consequence of A in $\mathcal{C}$ iff $\models_{\mathcal{C}} A \to B$

Satisfiability in a class of models

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

Given formula A, class of models $\mathcal{C}$: is A satisfiable in $\mathcal{C}$?

Examples

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

Proposition

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

The main reasoning problems

1. Model checking
   Given A, finite M, w in M: do we have $M, w \Vdash A$?

2. Satisfiability
   Given A, $\mathcal{C}$: is there $M \in \mathcal{C}$ and w in M s.th. $M, w \Vdash A$?

3. Model building
   Given A, $\mathcal{C}$:
   - if A is unsatisfiable in $\mathcal{C}$ then output "NO"
   - if A is satisfiable in $\mathcal{C}$ then
     output some model M that is in $\mathcal{C}$ and some w in M such that $M, w \Vdash A$

How can we solve them automatically?

Part 2: Talking about models

3 The modal language

4 Truth conditions

5 Reasoning in modal logics

6 The standard translation

The standard translation

Maps the language of modal logic to the language of FOL:

$$\begin{array}{rcll} ST(P, w) & = & P(w) & \\ ST(\neg A, w) & = & \neg ST(A, w) & \\ ST(A \wedge B, w) & = & ST(A, w) \wedge ST(B, w) & \\ ST(\Box_I A, w) & = & \forall u(R_I(w, u) \rightarrow ST(A, u) & \text{where } u \text{ is new} \\ ST(\Diamond_I A, w) & = & \exists u(R_I(w, u) \wedge ST(A, u)) & \text{where } u \text{ is new} \end{array}$$

Example

$ST(\Diamond_I \Diamond_J P) = \ldots$

Theorem

Suppose the class $\mathcal{C}$ can be defined by a FOL formula $A_{\mathcal{C}}$. Then A is $\mathcal{C}$-satisfiable iff $A_{\mathcal{C}} \wedge ST(A)$ is FOL-satisfiable.

The standard translation: examples

Examples

- The class of reflexive models KT is defined by the first-order formula $A_{\mathsf{KT}} = \forall w R(w, w)$.
- The class of transitive models K4 is defined by

$$A_{\mathsf{K4}} = \forall w_1 \forall w_2 \forall w_3 \ ((R(w_1, w_2) \wedge R(w_2, w_3)) \rightarrow R(w_1, w_3))$$

- The class of serial models is defined by ...
- The class of confluent models is defined by ...

Examples

- The class of finite models cannot be defined by a FOL formula.
- The class of models without infinite R-chains ('conversely well-founded') cannot be defined by a FOL formula.

The standard translation and the two-variable fragment of FOL

- FO2 = FOL with constants, equality and only two variables
  - satisfiability is decidable

    in nondeterministic exponential time (NEXPTIME)
- Standard translation with only two variables w_1 and w_1:

$$\begin{array}{lcl} ST(\Box_I A, w_1) & = & \forall w_2 (R_I(w_1, w_2) \rightarrow ST(A, w_2)) \\ ST(\Box_I A, w_2) & = & \forall w_1 (R_I(w_2, w_1) \rightarrow ST(A, w_1)) \end{array}$$

- Problem: to define transitivity in FOL we need three variables

  ... and FO3 is not decidable

Outline of course

Part 1: Modelling with graphs
Part 2: Talking about models
Part 3: The model construction method: basics
Part 4: Logics with simple constraints on models
Part 5: Logics with potential cycles
Part 6: Model checking in LoTREC
Part 7: Logics with transitive closure

Part 3:
The model construction method: basics

7 Outline of the method

8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K_n

Classical logic [Beth 55, Hintikka 55, Schütte 56; Smullyan 68]

Checking the satisfiability of a given formula A:

1. Try to find M and w by applying the 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 55, Hintikka 55, Schütte 56; Smullyan 68]

Checking the satisfiability of a given formula A:

1. Try to find M and w by applying the truth conditions
   - $M, w \Vdash A_1 \land A_2 \implies$ add $M, w \Vdash A_1$, and add $M, w \Vdash A_2$
   - $M, w \Vdash A_1 \lor A_2 \implies$ 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 $M, w \Vdash A_1$
     - $M, w \Vdash \neg(A_1 \lor A_2) \implies$ add $M, w \Vdash \neg A_1$ and add $M, w \Vdash \neg A_2$
     - $M, w \Vdash \neg(A_1 \land A_2) \implies$ 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 83]

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

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

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

$$\boxed{\texttt{[] P \& <> Q \& <> (R v \~{} P)}}$$

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)

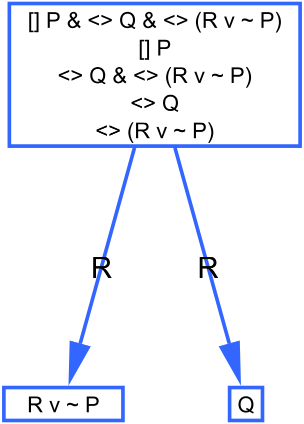
Example

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

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

Example

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



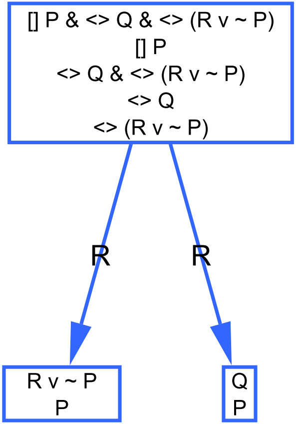
[] P & <> Q & <> (R v ~ P)
[] P
<> Q & <> (R v ~ P)
<> Q
<> (R v ~ P)

R v ~ P

Q

Example

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



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

R

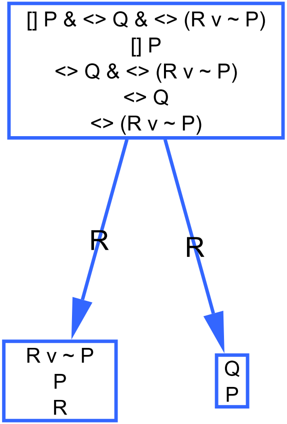
R

R v ~ P
P

Q
P

Example

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

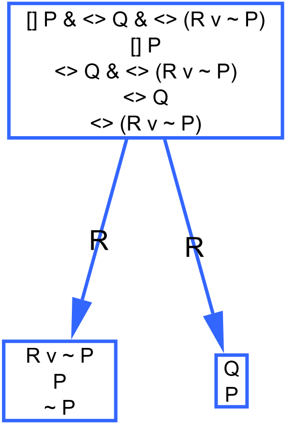


[] P & <> Q & <> (R v ~ P)
[] P
<> Q & <> (R v ~ P)
<> Q
<> (R v ~ P)

R v ~ P
P
R

Q
P

premodel 1



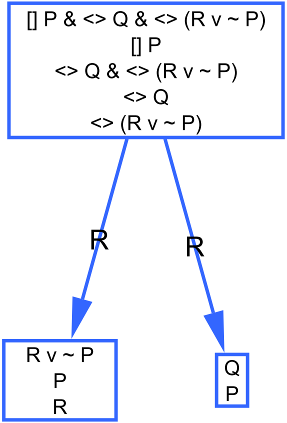
[] P & <> Q & <> (R v ~ P)
[] P
<> Q & <> (R v ~ P)
<> Q
<> (R v ~ P)

R v ~ P
P
~ P

Q
P

premodel 2

Example



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

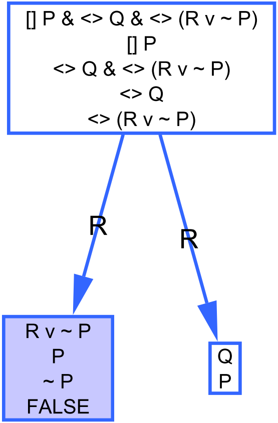
R

R

R v ~ P
P
R

Q
P

premodel 1



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

R

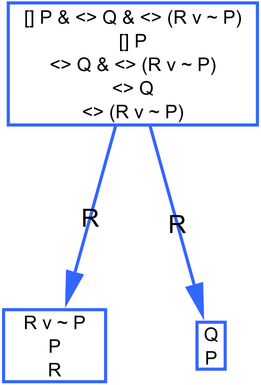
R

R v ~ P
P
~ P
FALSE

Q
P

premodel 2

Example



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

R

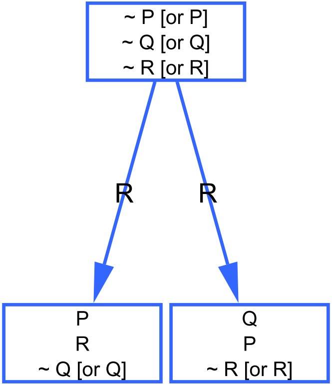
R

R v ~ P
P
R

Q
P

premodel 1

Model
$\Longrightarrow$
extraction



~ P [or P]
~ Q [or Q]
~ R [or R]

R

R

P
R
~ Q [or Q]

Q
P
~ R [or R]

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

A short history of the tableau method

Since 1950's: handwritten proofs

- . . . Sequent calculi [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. 97, Gasquet et al. 06]

Nowadays: automated provers

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

Part 3:
The model construction method: basics

7 Outline of the method

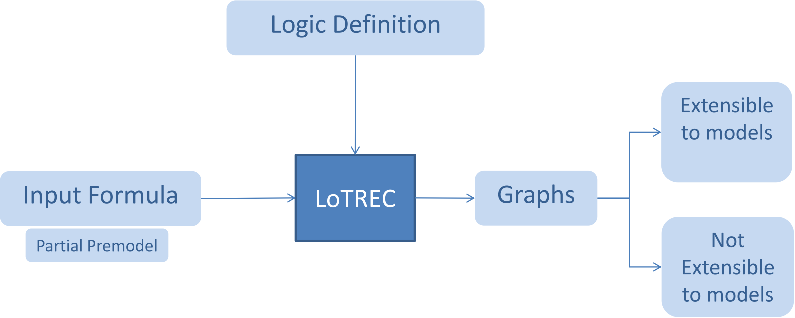
8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K_n

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 Said [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



Logic Definition

Input Formula

Partial Premodel

LoTREC

Graphs

Extensible to models

Not Extensible to models

Outline

7 Outline of the method

8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K_n

User-defined language

- Atomic propositions
  - constant symbols = 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

7 Outline of the method

8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

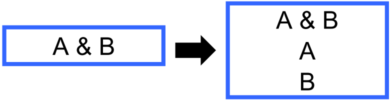
9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K_n

On paper

Truth conditions
$+$
Structural constraints

as Graph rewriting rules

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

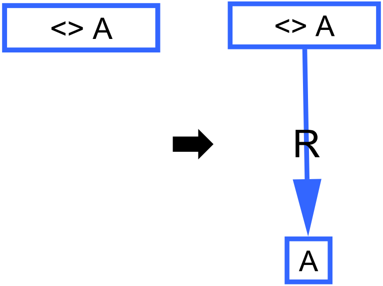


On paper

Truth conditions
+
Structural constraints

as Graph rewriting rules

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

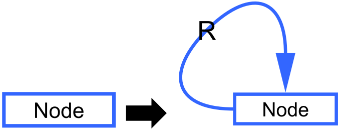


On paper

Truth conditions
+
Structural constraints

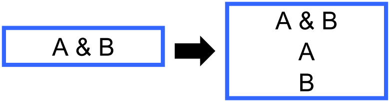
as Graph rewriting rules

Model is reflexive



In LoTREC

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



A & B

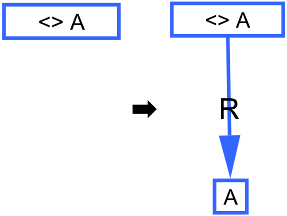
A & B
A
B

```
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"

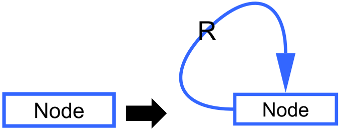


```
Rule Pos
 hasElement node1 pos variable A

 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

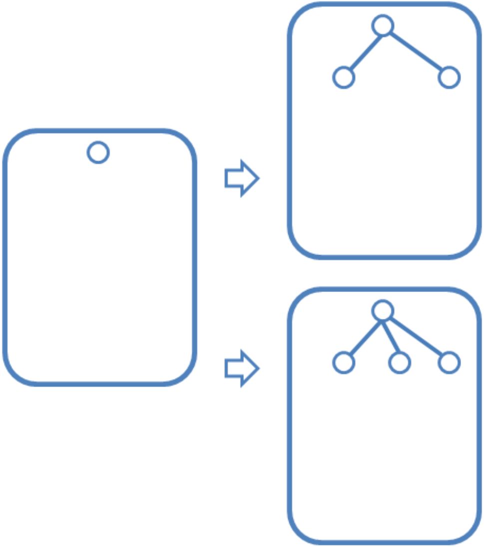
- adding links
- adding nodes
- adding formulas
- duplicating the graph

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

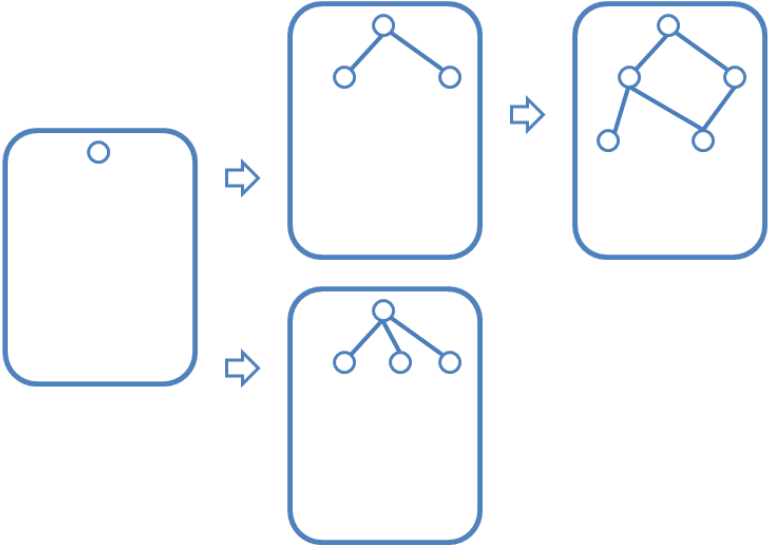
$$rule(\{G_1, \ldots, G_n\}) = rule(G_1) \cup \ldots \cup rule(G_n)$$

Managing graph copies: depth-first

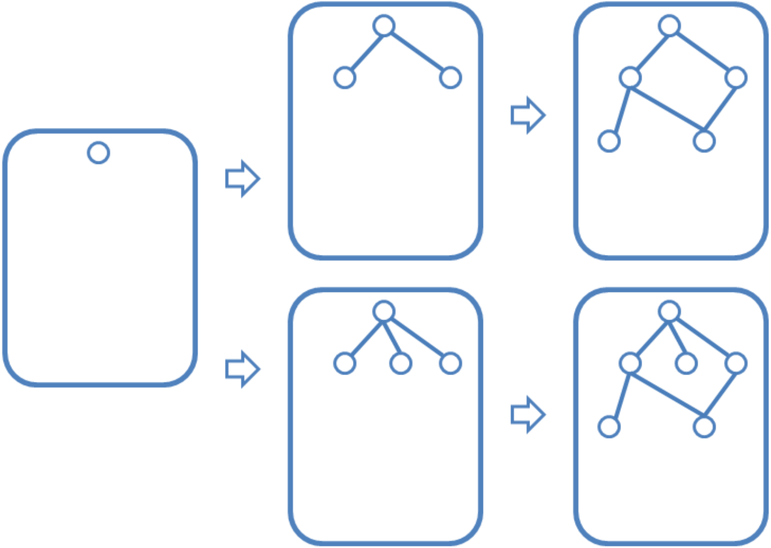
Managing graph copies: depth-first



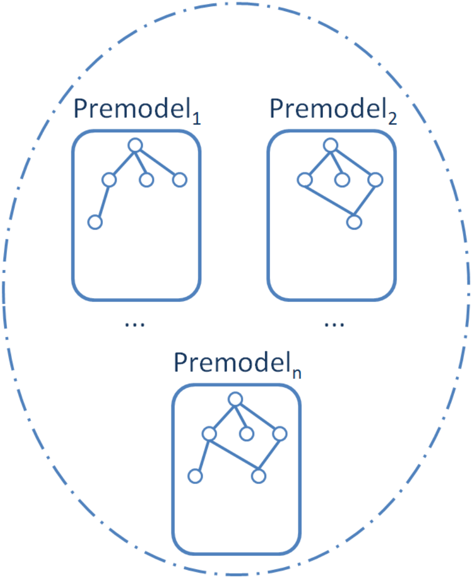
Managing graph copies: depth-first



Managing graph copies: depth-first



Managing graph copies: depth-first



Outline

Why a strategy?

- Apply rules in order:

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

- Saturation:

```
Strategy CPL_strat    Strategy K_strat
 repeat                 repeat
  Stop                   CPL
  NotNot                 Pos
   And                   Nec
   Or                   end
 end
```

Semantics of strategies

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

Example

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

Semantics of strategies

1. block: `rule1 ... rulen ... anotherStrategy ...`
   apply all applicable rules in order then stop
2. `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

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

Example

```
repeat
 firstRule
  rule1
  rule2    ✗
 end
end
```

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

Semantics of strategies

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

Example

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

Semantics of strategies

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

Tableaux: 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"
- 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

7 Outline of the method

8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K_n

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 tableau 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 tableau construction terminates

Part 3:
The model construction method: basics

7 Outline of the method

8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K_n

Outline

7 Outline of the method

8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K_n

How to proceed

CPL: Classical Propositional Logic

1. From the menu bar, open:
   $\Longrightarrow$ *Logic* $\Longrightarrow$ *Predefined logics* $\Longrightarrow$ *CPL*
2. Run: *Build Premodels* button
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 “$\leftrightarrow$” operator?

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

7 Outline of the method

8 Building models in LoTREC
  - Defining a language in LoTREC
  - Tableau rules
  - Strategies
  - Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
  - Classical logic
  - Modal logic K
  - Multi-modal logic K_n

From CPL to K

- Here: minimal set of operators $\neg$, $\wedge$, $\Box$ 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` operator
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

7 Outline of the method

8 Building models in LoTREC
- Defining a language in LoTREC
- Tableau rules
- Strategies
- Do the algorithms do the right thing?

9 The basic modal Logic and its implementation in LoTREC
- Classical logic
- Modal logic K
- Multi-modal logic K$_n$

From K To K_n

- Replace the operator $\Box_{\_}$ 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 operators 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 of course

Part 1: Modelling with graphs
Part 2: Talking about models
Part 3: The model construction method: basics
Part 4: Logics with simple constraints on models
Part 5: Logics with potential cycles
Part 6: Model checking in LoTREC
Part 7: Logics with transitive closure

Part 4:
Logics with simple constraints on models

10 KT
  - KD

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

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
```

$\Longrightarrow$ 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 $\Box P \land \Box \neg P \ldots \Longrightarrow$ 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 $\Box P \land \Box \neg P \ldots \Longrightarrow$ sound but suboptimal
- avoid too many successor nodes: apply `makeSerial` only once

```
   applyOnce makeSerial
```

Outline of course

Part 5: Logics with potential cycles

11 S4

12 Intuitionistic logic LJ

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. . .

Part 5: Logics with potential cycles

11 S4

12 Intuitionistic logic LJ

From S4 to intuitionistic logic LJ

- Accessibility relation R is reflexive, transitive, and *persistent*
- 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: $\neg\neg A \leftrightarrow A$; $\neg(A \land B) \leftrightarrow \neg A \lor \neg B$; …
  - tableau method requires *signed formulas*
  - in LoTREC: define operators `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 *persistent*
- 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 operators `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 *persistent*
- 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 operators `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 persistent R):

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

   add u sTrue variable a
```

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

Outline of course

- Part 1: Modelling with graphs
- Part 2: Talking about models
- Part 3: The model construction method: basics
- Part 4: Logics with simple constraints on models
- Part 5: Logics with potential cycles
- Part 6: Model checking in LoTREC
- Part 7: Logics with transitive closure

Part 6: Model checking in LoTREC

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 operator)

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 of course

Part 1: Modelling with graphs
Part 2: Talking about models
Part 3: The model construction method: basics
Part 4: Logics with simple constraints on models
Part 5: Logics with potential cycles
Part 6: Model checking in LoTREC
Part 7: Logics with transitive closure

Outline

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 $w R_\Pi w'$ 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

- PDL
- Suggestions

It is up to you...

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

Thank you!