# LoTREC: an environment for experimenting Kripke Semantics

Luis Fariñas del Cerro
Olivier Gasquet
Andreas Herzig
Mohamad Sahade

IRIT-CNRS
Toulouse

# Master in Artificial Intelligence

- Formalization of interaction between rational agents
- Belief-Desire-Intention framework
- Action and knowledge
- Temporal logics
- Description logics

# Modal Logics for Interaction

- Non-truth functional concepts
  - belief, time, action, obligation,…
- Modal connectives
  - $Bel_i$ A = "agent i believes that A"
  - F A = "A will be true at some future time point"
  - $After_a$ A = "A is true after action a"
  - $Oblg_i$ A = "A is obligatory for agent i"
- Generic form:
  - []A = "A is true in all possible worlds"
  - <>A = "A is true in some possible world"

# Logics of action and knowledge

- modal operators

  $Knw_i\ A$ "agent i knows that A"

  $[a]\ A$ "after execution of action a, A holds"

- axiomatically:

  $Knw_i[a]A \leftrightarrow [a]Knw_iA$

  $<a>Knw_iA \rightarrow Knw_i<a>A$

- relational properties:

  $R_{knwi} \circ R_a = R_a \circ R_{knwi}$ (permutation)

  $(R_{knwi})^{-1} \circ R_a = R_a \circ (R_{knwi})^{-1}$ (confluence)

# Belief-Desire-Intention logics

- Modal operators

| | |
|---|---|
| $Bel_i$ A | "agent i believes that A" |
| $Desire_i$ A | "agent i desires that A" |
| $Intend_i$ A | "agent i intends that A" |

- Branching temporal logic

# Interpreting the language: truth conditions

- Models = Worlds + Relations
- classical connectives

  M,w ||- P  iff $V_w(P) = 1$, for P in Atoms

  M,w ||- A∧B  iff (M,w ||- A and M,w ||- B)
- non-classical connectives
  - interpretation via accessibility relation R
- the basic modal operators:

  M,w ||- []A  iff forall u: Rwu implies M,u ||- A

  M,w ||- <>A  iff exists u: Rwu and M,u ||- A

# Models

- model M = (W,R,V)
  - W nonempty set (possible worlds)
  - R: Ops $\rightarrow$ (WxW) (accessibility relation)
  - V: W $\rightarrow$ (Atoms $\rightarrow$ {0,1}) (valuation)
- pointed model ((W,R,V),w)
  - w in W (actual world)

# Validity and satisfiability in a class of models *Cls*

- *Cls* some subset of **K** (one for each logic)
- A is *valid* in *Cls* iff A is true at any w of any M in *Cls* ($\models_{Cls}$ A)

  examples: $\Box P \rightarrow P$ invalid in **K**
  $\Box P \rightarrow P$ valid in the class of reflexive models
  $\Diamond P \rightarrow \Diamond\Diamond P$ valid in transitive models

- A is *satisfiable* in *Cls* iff A is true at some w of some M in *Cls*

  examples: $P \wedge \sim\Box P$ satisfiable in K
  $P \wedge \sim\Box P$ unsatisfiable in reflexive models

A is valid in *Cls* iff ~A is unsatisfiable in *Cls*

# Reasoning problems

- model checking

  given A, M and w, do we have M,w ||- A?

- validity

  given A and Cls, is A valid in Cls?

- satisfiability

  given A and Cls, does there exist M in Cls and w in M such that M,w ||- A?

# Reasoning problems

- model checking

  given A, M and w, do we have M,w ||- A?

- validity

  given A and Cls, is A valid in Cls?

- satisfiability

  given A and Cls, does there exist M in Cls and w in M such that M,w ||- A?

How can we solve them automatically?

# Existing

- Optimized tableaux provers (LWB,FaCT):
  - hacking skills needed (C++,...)
  - do not provide models (yes/no answer)
  - use tricky optimizations (Backjumping)
- no generic natural deduction system
- Hilbert-style proofs are useless
- translation methods difficult to read and need specialized skolemization

# Needs

- Build Cls-models (= build graphs)
- Complete premodels to Cls-models
- Evaluate modal formulas on some model
- Look for Cls-models of a formula
- No high programming skills
- Basics on graphs and propositional logic

# Overview

- LoTREC: building Kripke models
- tableaux systems: basic ideas
- tableaux systems: basic definitions
- tableaux for simple modal logics
- tableaux for transitive modal logics
- termination, soundness and completeness

# LoTREC

- IRIT-CNRS Toulouse
- Downloadable from www.irit.fr
- Explicit accessibility relations
- Easy implementation of complex logics (symmetry, linearity, transitivity, PDL, DL with cardinal restrictions...)
- ... and inefficient

# Overview

- LoTREC: building Kripke models
- **tableaux systems: basic ideas**
- tableaux systems: basic definitions
- tableaux for simple modal logics
- tableaux for transitive modal logics
- soundness and completeness

# The basic idea for modal logics

- apply truth conditions and guaranty relational properties = build a graph
  - create nodes
  - add links between nodes
  - add formulas to node

- the basic modal cases
  - M,w ||- []A ➔ forall u such that Rwu, add u ||- A
  - M,w ||- <>A ➔ add some new u, add Rwu, add u ||- A

# The basic idea: example for modal logic

Find a reflexive model for A = P $\wedge$ <>(~P$\wedge$[]P)

- applying truth conditions:
  - ➔ M,w ||- P$\wedge$<>(~P$\wedge$[]P)

# The basic idea: example for modal logic

Find a reflexive model for A = P $\wedge$ <>(~P$\wedge$[]P)

- applying truth conditions:
  - M,w ||- P$\wedge$<>(~P$\wedge$[]P)
  - M,w ||- P  M,w ||- <>(~P$\wedge$[]P)

# The basic idea: example for modal logic

Find a reflexive model for A = P $\wedge$ <>(~P$\wedge$[]P)

- applying truth conditions:
  - M,w ||- P$\wedge$<>(~P$\wedge$[]P)
  - M,w ||- P  M,w ||- <>(~P$\wedge$[]P)

# The basic idea: example for modal logic

Find a reflexive model for A = P ∧ <>(~P∧[]P)

- applying truth conditions:
  - M,w ||- P∧<>(~P∧[]P)
  - M,w ||- P  M,w ||- <>(~P∧[]P)
  - New u : Rwu, M,u ||- (~P∧[]P)

# The basic idea: example for modal logic

Find a reflexive model for A = P ∧ <>(~P∧[]P)

- applying truth conditions:
  - M,w ||- P∧<>(~P∧[]P)
  - M,w ||- P  M,w ||- <>(~P∧[]P)
  - New u : Rwu, M,u ||- (~P∧[]P)
  - M,u ||- ~P and M,u ||- []P

# The basic idea: example for modal logic

Find a reflexive model for A = P $\wedge$ <>(~P$\wedge$[]P)

- applying truth conditions:
  - M,w ||- P$\wedge$<>(~P$\wedge$[]P)
  - M,w ||- P  M,w ||- <>(~P$\wedge$[]P)
  - New u : Rwu, M,u ||- (~P$\wedge$[]P)
  - M,u ||- ~P and M,u ||- []P

No more truth conditions
But not reflexive model

# The basic idea: example for modal logic

Find a reflexive model for A = P $\wedge$ <>(~P$\wedge$[]P)

- applying truth conditions:
  - → M,w ||- P$\wedge$<>(~P$\wedge$[]P)
  - → M,w ||- P  M,w ||- <>(~P$\wedge$[]P)
  - → New u : Rwu, M,u ||- (~P$\wedge$[]P)
  - → M,u ||- ~P and M,u ||- []P
- applying completion: Rww and Ruu

# The basic idea: example for modal logic

Find a reflexive model for A = P ∧ <>(~P∧[]P)

- applying truth conditions:
  - → M,w ||- P∧<>(~P∧[]P)
  - → M,w ||- P  M,w ||- <>(~P∧[]P)
  - → New u : Rwu, M,u ||- (~P∧[]P)
  - → M,u ||- ~P and M,u ||- []P
- applying completion: Rww and Ruu

But then apply [] truth condition on u

# The basic idea: example for modal logic

Find a reflexive model for A = P $\wedge$ <>(~P$\wedge$[]P)

- applying truth conditions:
  - M,w ||- P$\wedge$<>(~P$\wedge$[]P)
  - M,w ||- P  M,w ||- <>(~P$\wedge$[]P)
  - New u : Rwu, M,u ||- (~P$\wedge$[]P)
  - M,u ||- ~P and M,u ||- []P
- applying completion: Rww and Ruu
- applying again truth conditions:
  - M,u ||- P  Contradiction !

# The basic idea: example for modal logic

Find a reflexive model for A = P $\wedge$ <>(~P$\wedge$[]Q)

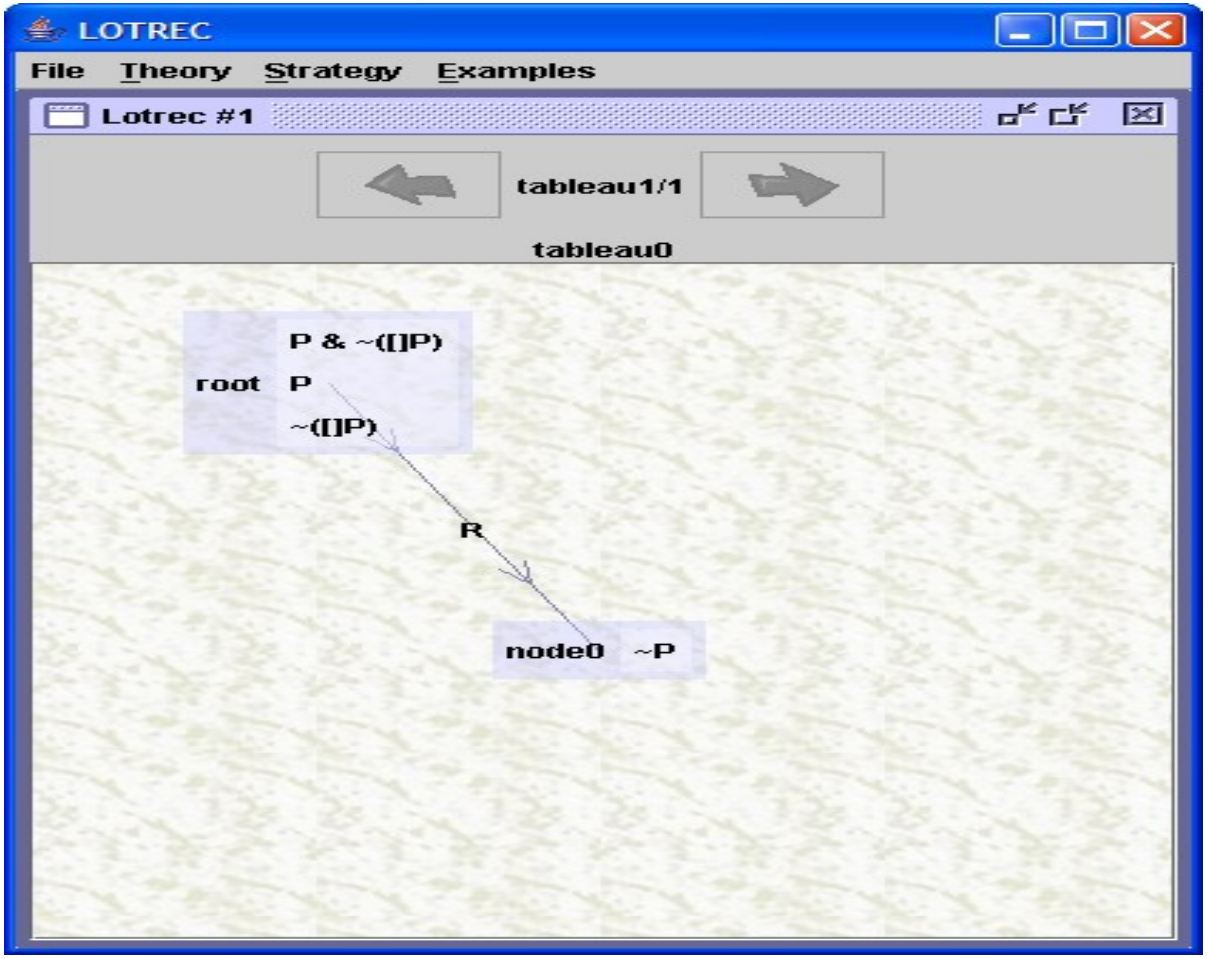
- applying truth conditions:
  - M,w ||- P  M,w ||- <>(~P$\wedge$[]Q)
  - Rwu and M,u ||- ~P and M,u ||- []Q
- applying completion:
  - Rww and Ruu
- applying again truth conditions:
  - M,u ||- Q  Reflexive model of A

# The basic idea: example for modal logic (ctd.)

- premodel for
  $A = P \wedge \langle\rangle\sim P$

➔ not closed

➔ model of A



# Overview

- LoTREC: building Kripke models
- tableaux systems: basic ideas
- **tableaux systems: basic definitions**
- tableaux for simple modal logics
- tableaux for transitive modal logics
- soundness and completeness

# Informal definition of tableau rules

- Tableau rules: truth conditions and relational properties by expanding graphs.
- The needs are
  - add formulas to nodes
  - add nodes
  - add  links
  - duplicate the graph (for disjunctive formulas/properties)
  - define strategies (ordering of rules)

# Informal definition of tableau rules

- Tableau rules: truth conditions and relational properties by expanding graphs.

LoTREC: apply rule to *every* pattern of G

➔ strategies get more declarative
➔ programming is easy
➔ proofs get easier

# Tableau rules: syntax

- general form:

  rule *ruleName*

  if $cond_1$

  …

  if $cond_n$

  do $action_1$

  …

  do $action_k$

# Tableau rules: syntax

- general form:

  rule *ruleName*
  if $cond_1$
  …
  if $cond_n$
  do $action_1$
  …
  do $action_k$

- example conditions:

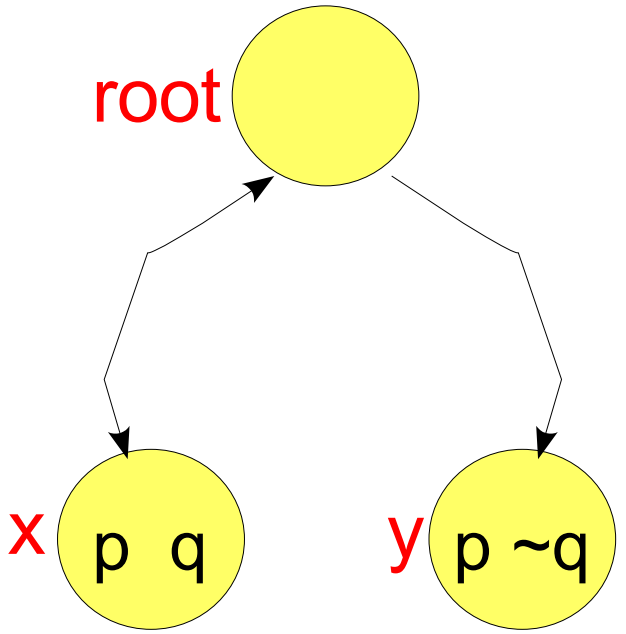
  if hasElement node formula
  if isLinked $node_1$ $node_2$ R
  *... (more to come)*

- example actions:

  do stop
  do add node formula
  do newNode node
  do link $node_1$ $node_2$ R
  do duplicate $node_1$ […]
  *... (more to come)*

# Building model is easy

```
rule model1
    if isNewNode w
    do newNode x
    do newNode y
    do link w x R
    do link x w R
    do link w y R
    do add x constant p
    do add x  constant q
    do add y  constant p
    do add y (not  constant q)
end
```



# Definition of strategies

- A *strategy* defines some order of application of the tableau rules:

  firstrule $rule_1$ … $rule_n$ end

  "apply first applicable rule (and stop)"

  allrules $rule_1$ … $rule_n$ end

  "apply all applicable rules in order"

  repeat *strategy* end

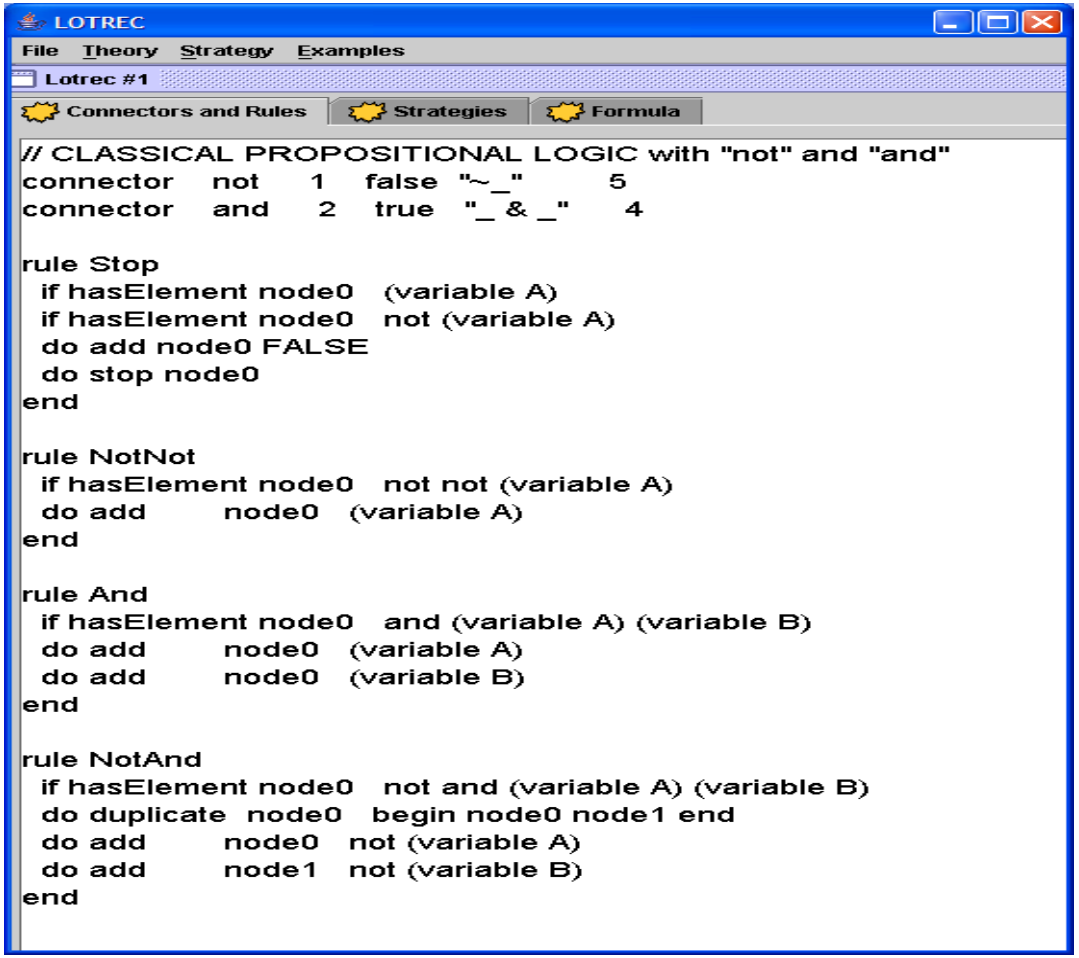
  "repeat until no rule applicable"

- Strategy stops if no rule is applicable.

# Strategy for classical logic

strategy CPLStrategy
repeat allRules
  Stop
  NotNot
  And
  NotAnd
end end
end

➔ “fair strategy”



# Definition of tableaux

The *set of tableaux for 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)

# Tableaux: 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.

# Formal properties

To be proved for each strategy:

- 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 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

# Overview

- LoTREC: building Kripke models
- tableaux systems: basic ideas
- tableaux systems: basic definitions
- **tableaux for simple modal logics**
- tableaux for transitive modal logics
- soundness and completeness

# Tableau rules for K

not, and, nec

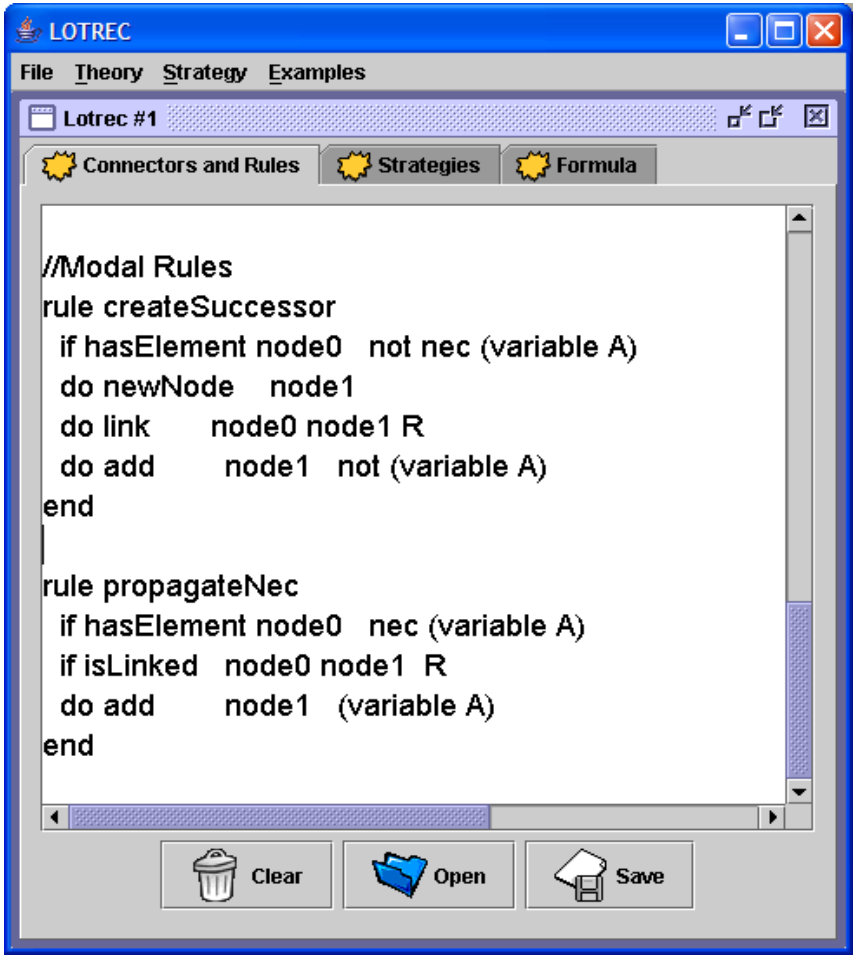
*[rules for classical logic]*

# Tableau rules for K

*not, and, nec*

*[rules for classical logic]*

createSuccessor:
  if ~[]A is in node0
  then
      create a new node1
      link it to node0
      add A to node1
end

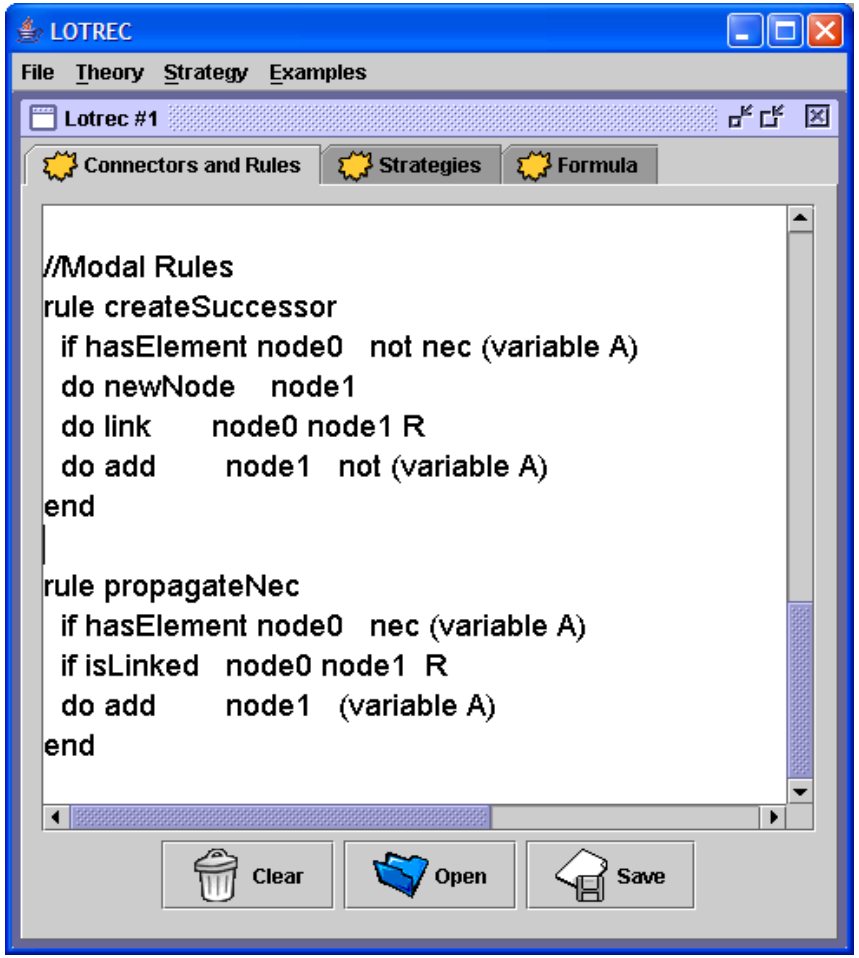


# Tableau rules for K

*not, and, nec*

*[rules for classical logic]*

propagateNec:
if []A is in node0
   node0 is linked to node1
then add  A to node1
end



# Modal logic KT

- accessibility relation is *reflexive*
- alternative way: integrate this into truth condition
  - M,w ||- []A  iff  forall u: Rwu implies M,u ||- A

# Modal logic KT

- accessibility relation is *reflexive*
- alternative way: integrate this into truth condition
  - M,w ||- []A  iff  forall u: Rwu implies M,u ||- A
    **and M,w ||- A**

# Tableaux
# for modal logic KT

*[connectors as for K…]*

*[rules as for K…]*

# Tableaux for modal logic KT

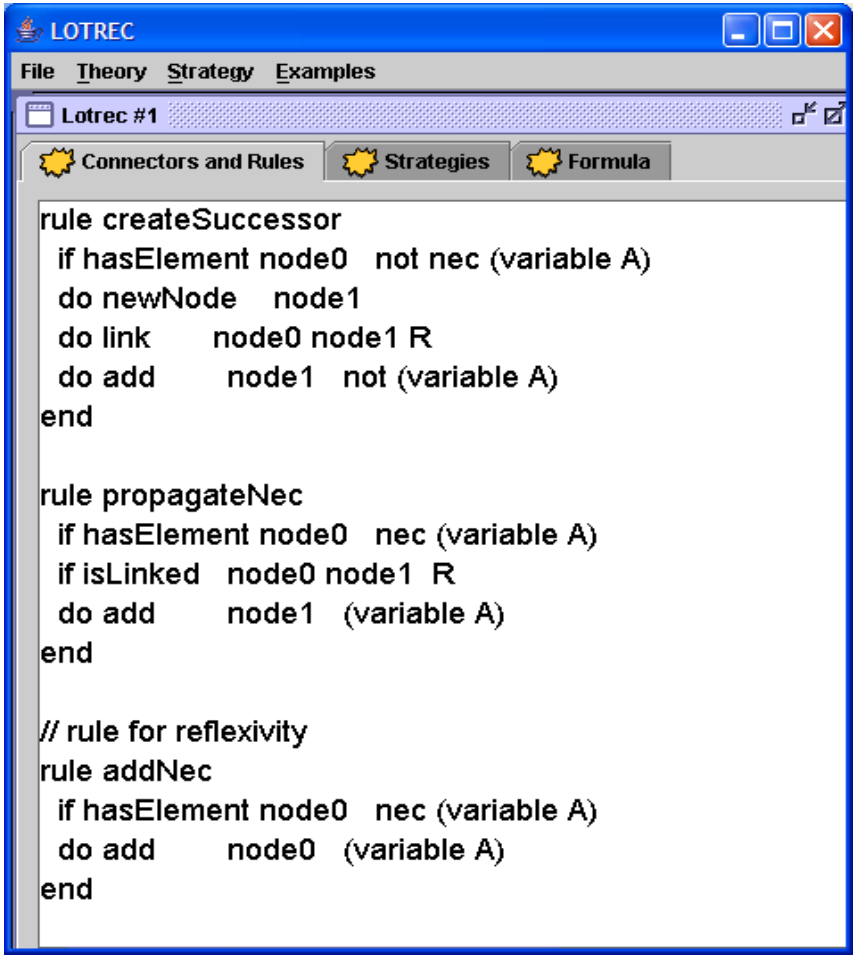
*[connectors as for K…]*

*[rules as for K…]*

plus:

"when []A is in a node
then add A to it"

KTStrategy(P & [][]~P)



# Tableaux for modal logic S5

accessibility relation is
equivalence relation

can be supposed to be
a single equivalence
class

optimized tableau rules
…

# Overview

- LoTREC: building Kripke models
- tableaux systems: basic ideas
- tableaux systems: basic definitions
- tableaux for simple modal logics
- **tableaux for transitive modal logics**
- soundness and completeness

# Tableau rules for S4
## (Advanced topics)

accessibility relation is *reflexive* and *transitive*

tableau rules for S4:

- *[connectors as for KT…]*
- *[rules as for KT…]*
- … and take into account transitivity:
  either explicitely
  or by
  if []A is in a node
  then add []A to all children nodes

# Tableau rules for S4
## (Advanced topics)

accessibility relation is *reflexive* and *transitive*

tableau rules for S4:

- *[connectors as for KT…]*
- *[rules as for KT…]*
- … and take into account transitivity:

if []A is in a node
then add []A to all children nodes”

problem: find a terminating strategy

# Tableau rules for S4

- Example: M,w ||- []~[]P
  - add ~[]P in w (by rule for reflexivity)

# Tableau rules for S4

- Example: M,w ||- []~[]P
  - add ~[]P in w (by rule for reflexivity)
  - create u, add Rwu, add ~P in u (by createSuccessor)

# Tableau rules for S4

- Example: M,w ||- []~[]P
  - add ~[]P in w (by rule for reflexivity)
  - create u, add Rwu, add ~P in u (by createSuccessor)
  - add []~[]P in u (by rule for transitivity)

# Tableau rules for S4

- Example: M,w ||- []~[]P
  - add ~[]P in w (by rule for reflexivity)
  - create u, add Rwu, add ~P in u (by createSuccessor)
  - add []~[]P in u (by rule for transitivity)
  - add ~[]P in u (by rule for reflexivity)

# Tableau rules for S4

- Example: M,w ||- []~[]P
  - add ~[]P in w (by rule for reflexivity)
  - create u, add Rwu, add ~P in u (by createSuccessor)
  - add []~[]P in u (by rule for transitivity)
  - add ~[]P in u (by rule for reflexivity)
  - create u'
  - …

# Tableau rules for S4

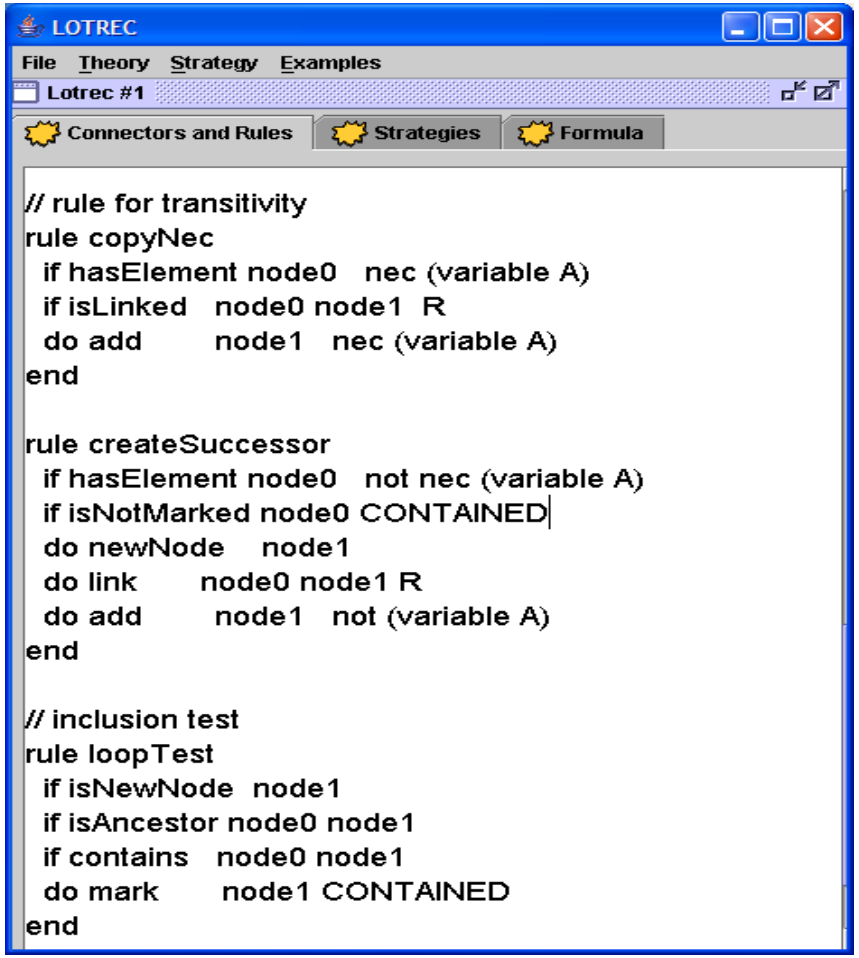
- Example: M,w ||- []~[]P
  - add ~[]P in w (by rule for reflexivity)
  - create u, add Rwu, add ~P in u (by createSuccessor)
  - add []~[]P in u (by rule for transitivity)
  - add ~[]P in u (by rule for reflexivity)
  - create u'
  - …

put a looptest into the rules!

# Tableau rules for S4 (ctd.)

principle:

- if a node is *included* in an ancestor
  then mark it.



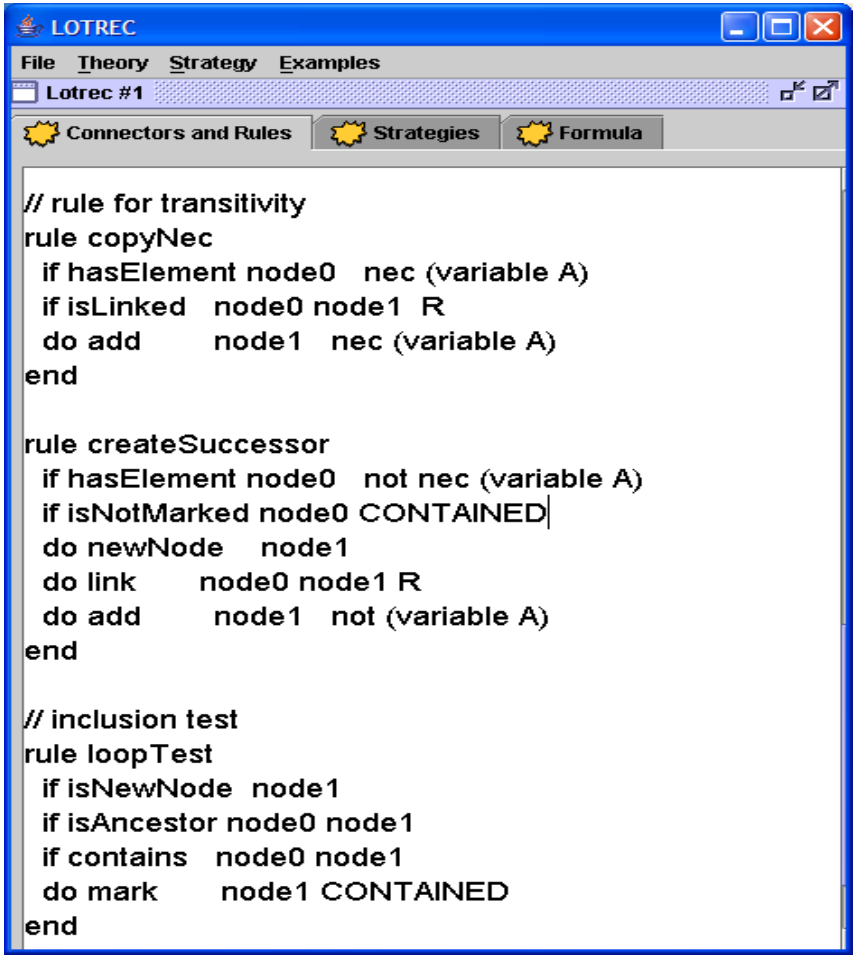
# Tableau rules for S4 (ctd.)

principle:

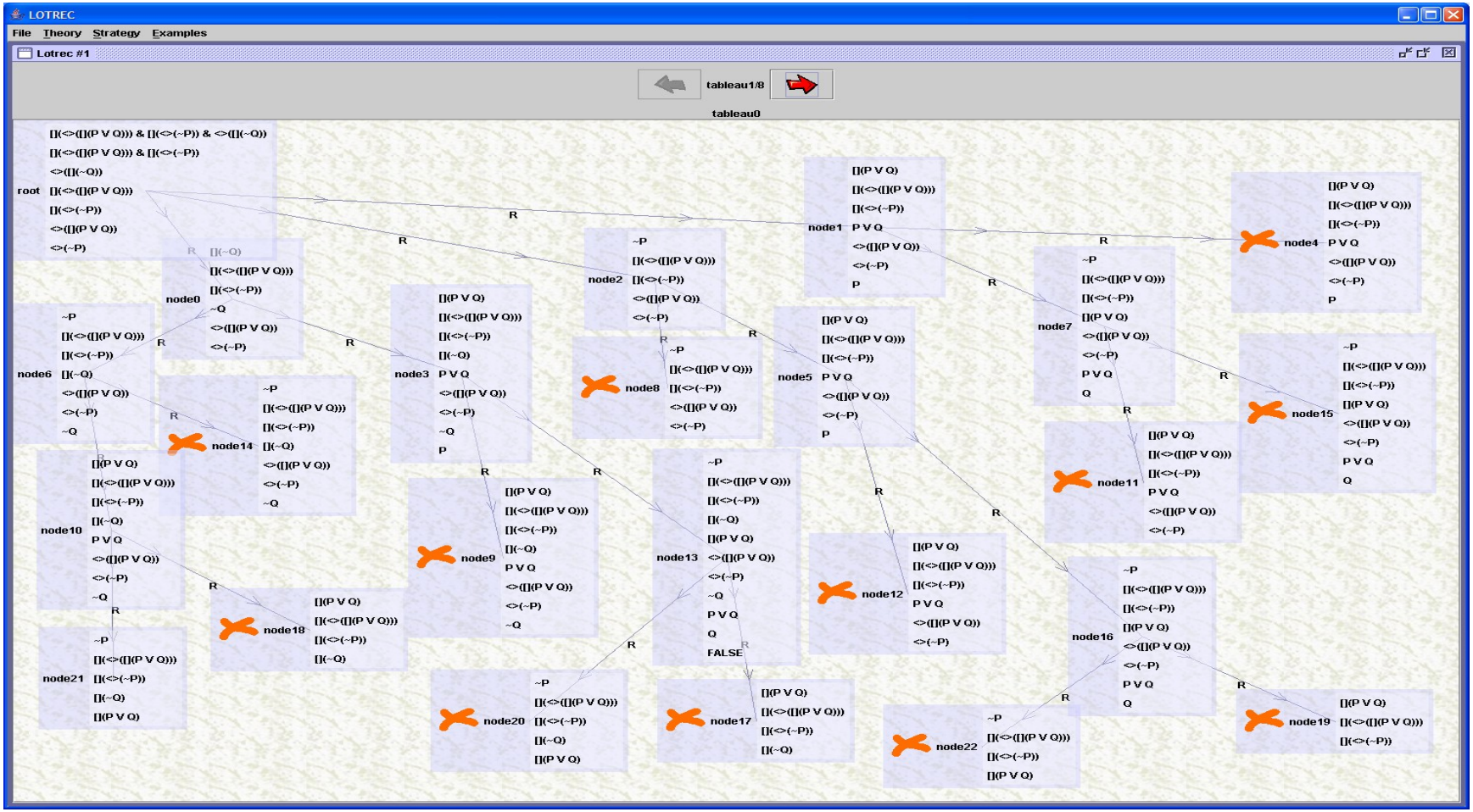
- if a node is *included* in an ancestor
  then mark it.

- if a node is marked
  then block the createSuccessor rule

- S4Strategy([]~[]P)



# S4Strategy
# ([]<>[] (P v Q) & []<>~P & <>[]~Q)



# Overview

- LoTREC: building Kripke models
- tableaux systems: basic ideas
- tableaux systems: basic definitions
- tableaux for simple modal logics
- tableaux for transitive modal logics
- **termination, soundness and completeness**

# A general termination theorem

- IF for every rule r:
  (the do-part of r only contains **strict** subformulas of the if-part of r
  AND
  [some restriction on node creation])

- THEN
  for every formula A:
  the tableaux construction terminates

# Another general termination theorem

- IF for every rule r:
  the do-part of r only contains subformulas of the if-part of r
  AND
  [some restriction on node creation]
  AND
  [some *looptesting* in the strategy]
- THEN
  for every formula A:
  the tableaux construction terminates

# Soundness

- Let L be any logic in {….}. If L-Strategy(A) is closed then A is L-unsatisfiable.

- Proof:
  - Every tableau rule is “guaranteed” by the truth conditions:
    If G is L-satisfiable
    then there is $G_i$ in rule(G) that is L-satisfiable
  - Hence if *every* tableau is closed
    then the original A cannot be L-satisfiable.

# Completeness

- Let L be any logic in {….}. If L-Strategy(A) is open then A is L-satisfiable.
- Proof:
  - Take some open tableau G in LStrategy(A).
  - Build a L-model from G:

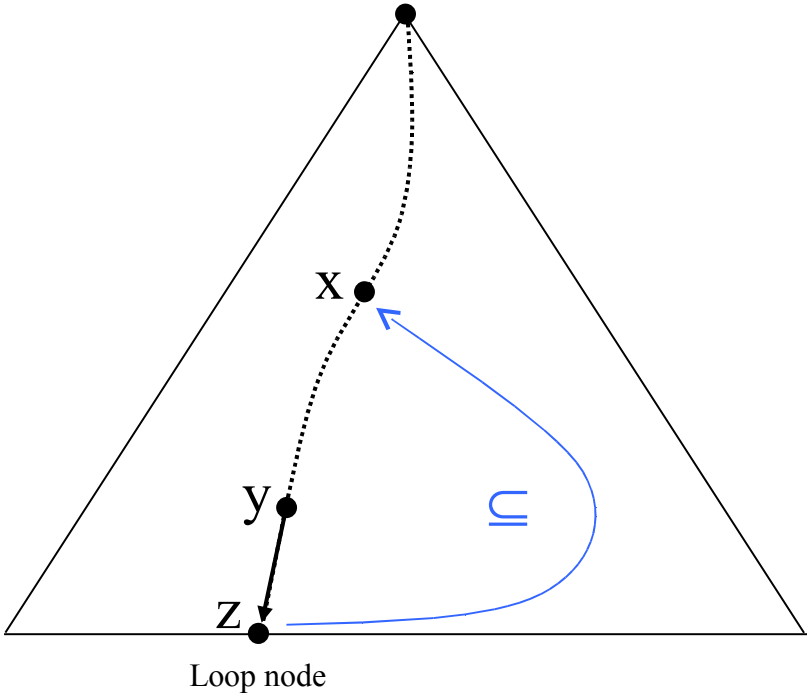
    $V_{node}(P) = 1$ iff P appears in node and P atomic
  - Prove by induction on the form of A:

    for every A in node, M,node ||- A

    ("fundamental lemma")
  - Folding…

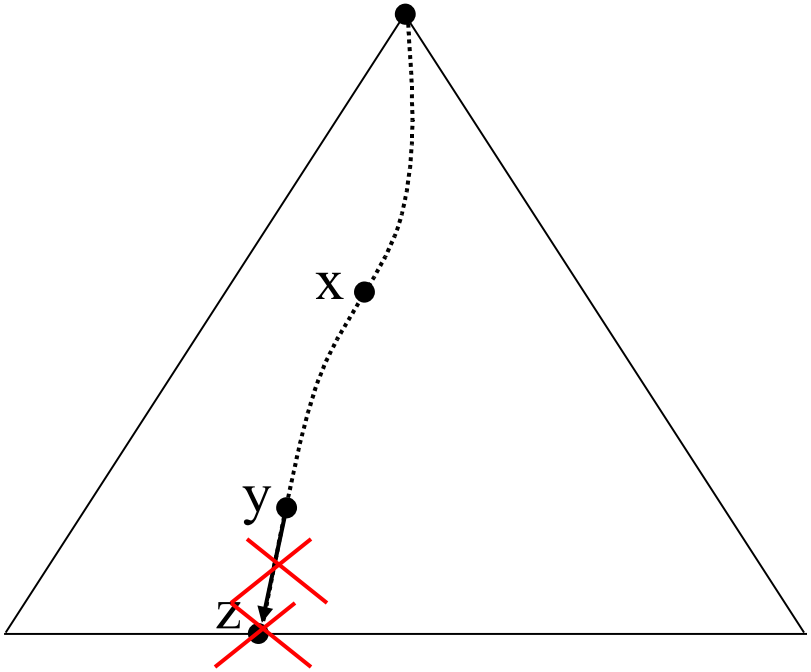
# Folding

1. Loop detection



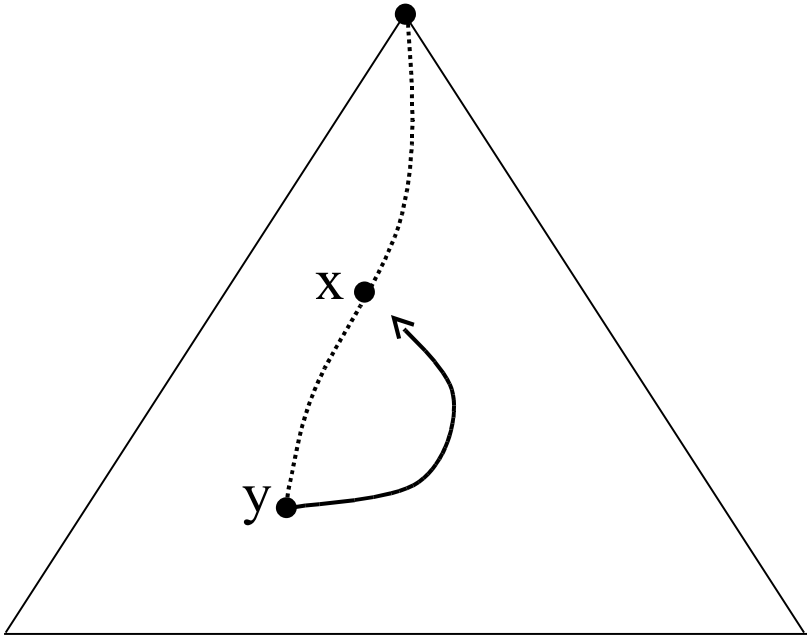
# Folding

2. Node and edge deletion



# Folding

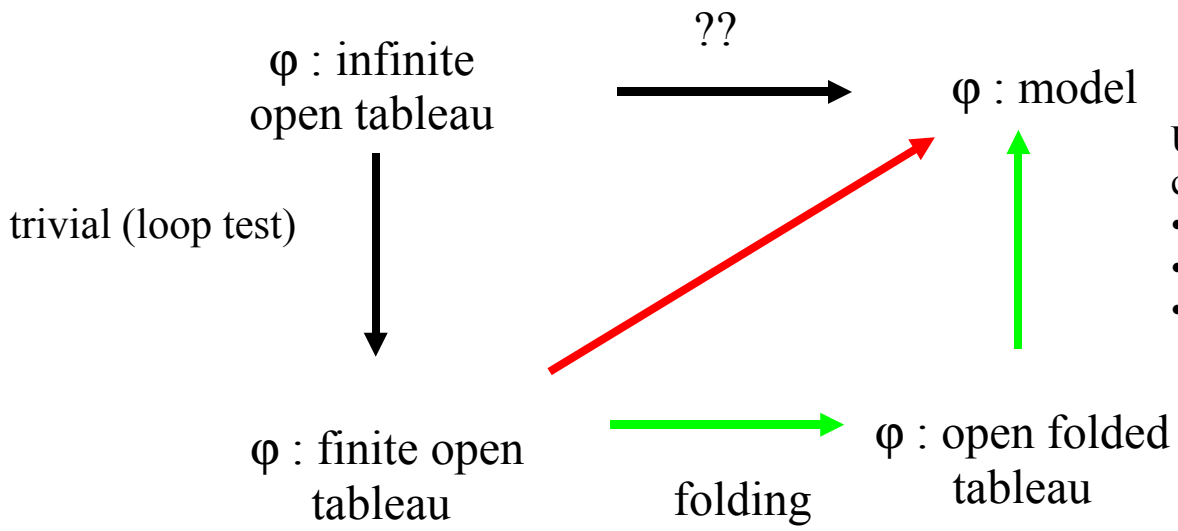
3. Folding itself



# Completeness

Theorem :

**ANY** FAIR STRATEGY for a logic K+[D, T,4, 5, B ] using the adequate rules is terminating and complete.




Usual proofs (when not only claimed without proof) are:
- Very complex;
- Imperative;
- Strategy-dependent.

# Future improvements

- Non polish notation for input formulas
- Drag-and-drop model building
- Drag-and-drop model modification
- Graphical design of rules
- Reuse of models
- First-order logic
- Interface with efficient SAT solvers
- Nodes collapsing, ...

# LoTREC on LILaC homepage at www.irit.fr (~ 6.5 Mo)

- model construction
  = model description
- formula evalutation and model search
  = description of truth conditions
  and relational properties
- wide range of logics:
  multiK +/- {F,D,T,B,4,5,Lin,Cf,...}
  LTL, PDL,DL's, some non-normal ML's
- correctness and termination results