CTL for Testing

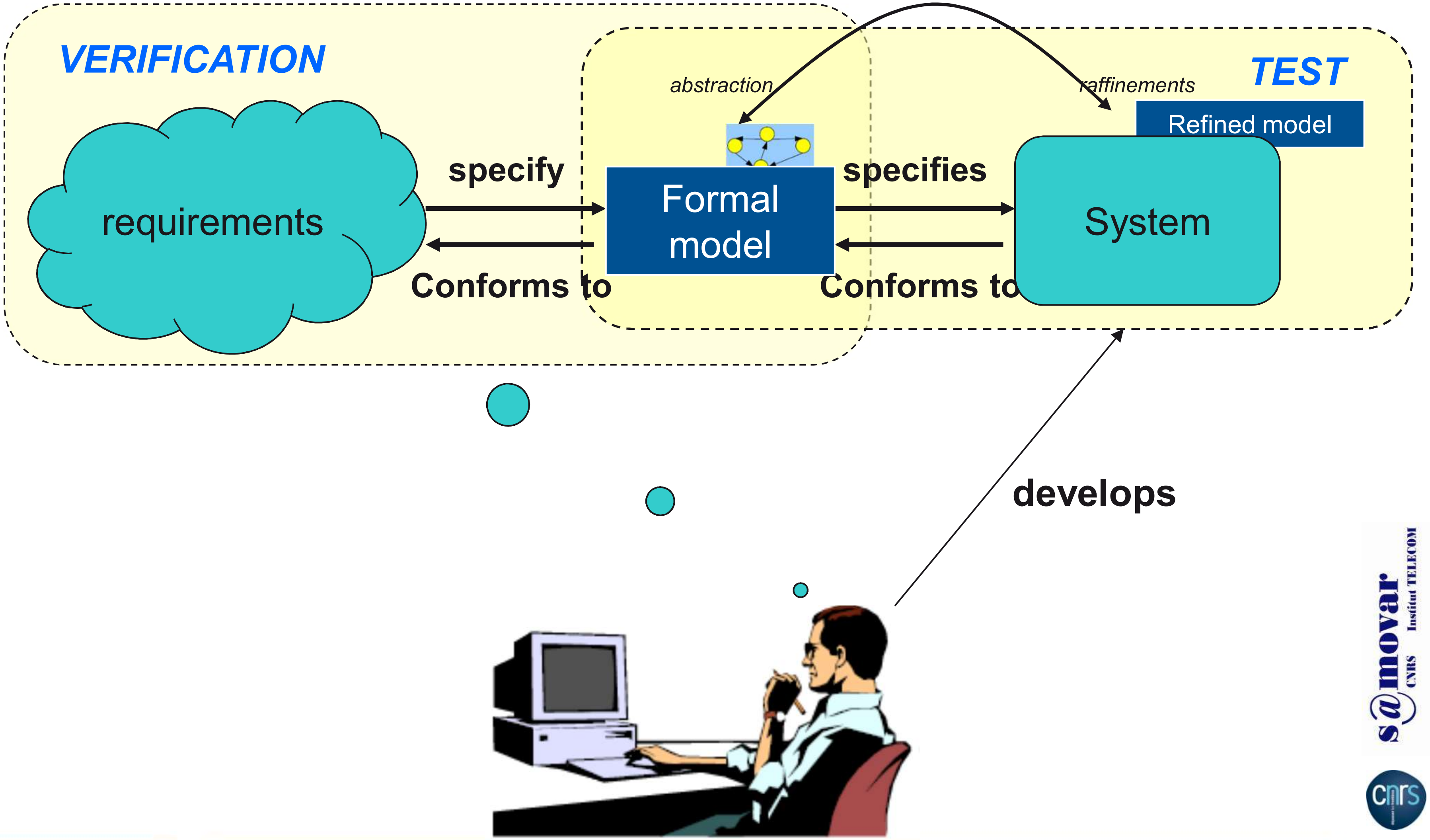
Basics

Stephane Maag

CNRS Samovar

Stephane.Maag@telecom-sudparis.eu

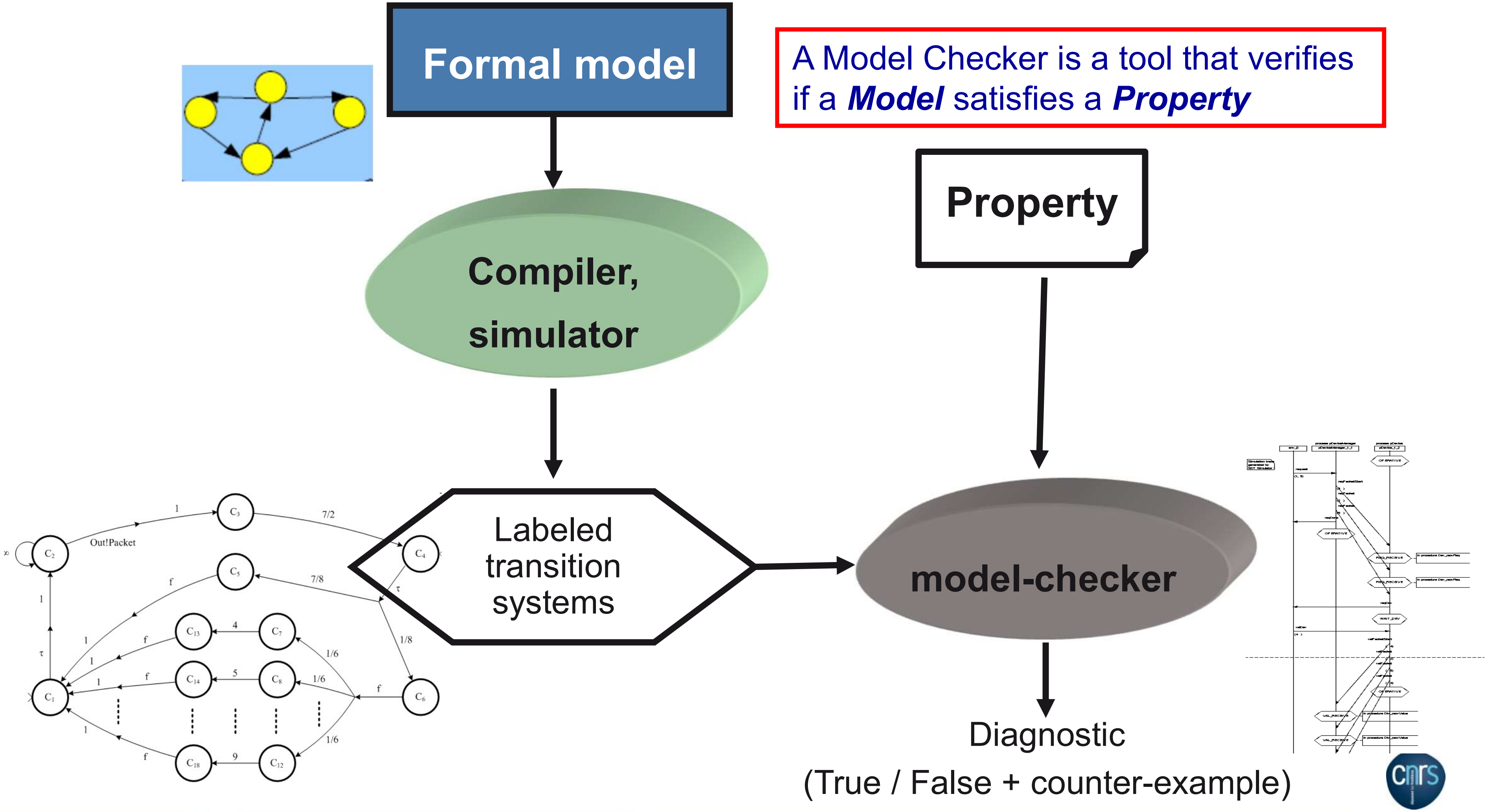
Flash back – reminder …



Formal verification techniques

- **3 main techniques**
  - **Code verification**
    - Static analysis – no formal models
    - Reverse engineering
    - BLAST, SLAM for C prog.
    - Bandera: JAVA
    - Verisoft: C++
  - 3 kinds of methods
    - Model equivalences
    - Deductive methods (proof)
    - Model checking

Model-checking Basis



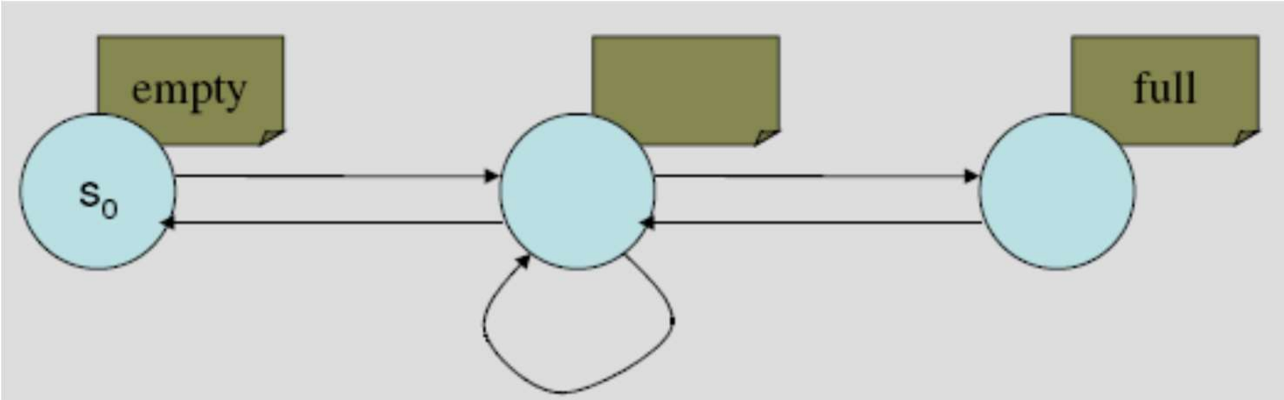
Model: a term with so many meanings !

- Here models – as they are used for model-checking are just *annotated graphs:*
  - A finite set of states, S
  - Some initial state s_0
  - A transition relation between states, $T \subseteq S \times S$
  - A finite set of atomic propositions, AP
  - A labelling function $L : S \rightarrow P(AP)$

- known as a Kripke structure:
  - Labelled Transition systems, LTS
  - Finite State machines, FSM
  - State charts, …

** For a physicist a "model" is a differential equation; For a biologist, it may be … mice or frogs*

An Example



AP = {empty, full}

Some LTL formula that are valid for this model:

empty ⇒ (X ¬empty)

full ⇒ (X ¬full)

(X is for neXt)

Systems are the actual objects of interest

- **How to ensure that a system satisfies certain properties?**
  - But what are *properties ?!*
- Properties?

1. Texts in natural languages…

   “**Calls to** lock **and** unlock **must** alternate**.**”

2. Formulas in a given specification logic

   (locked $\Rightarrow$ X unlocked) $\wedge$ (unlocked $\Rightarrow$ X locked)

3. Sets of mandatory or forbidden behaviors

Kinds of functional properties

Reachability	**A state in a system may be reached**
	The train may cross the railroad crossing
Liveness	**Under some conditions, an event will come**
	When the train announced its arrival, the gate is closed
Safety	**A non desired event will never occur**
	It is not possible to have the gate open while the train cross the railroad crossing.
No deadlock	**The system will never reach a state from which it can not evolve anymore.**
	When the gate is closed, it can still be opened.
Fairness	**An event will occur indefinitely often**
	The gate will be open indefinitely often.

Model-checking problem

I |= S

System model → I

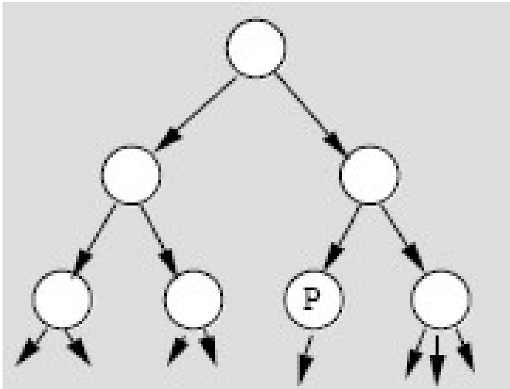
|= ← Satisfaction relation

S ← System property

The CTL Logic
Computation Tree Logic

- CTL allows to reason on computation tree

Examples



There exists a path with a state in which P holds

EF P

Temporal operators on an execution : X, F, G, U

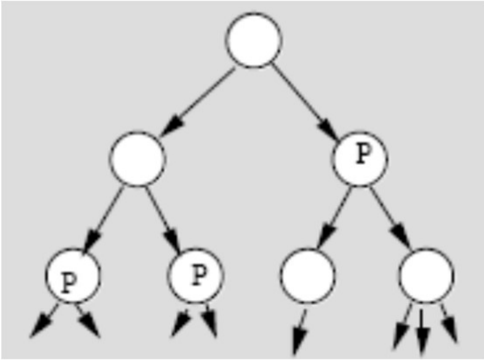
- X φ : the next state satisfies φ (neXt)
- F φ : there exists a state in the future which satisfies φ (Future)
- G φ : all the states satisfy φ (Global)

  $G\ \varphi\ (= \neg F\ \neg\varphi)$

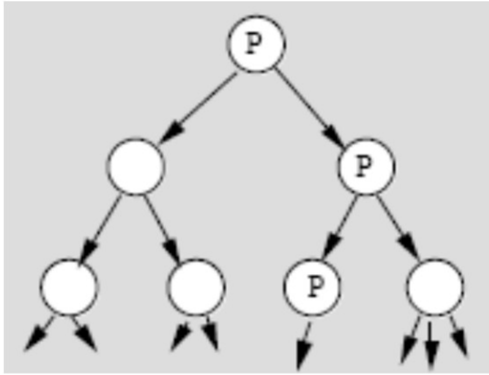
- φ U Ψ : a state in which Ψ holds and up to this state φ holds true (Until)

  $F\ \Psi \Leftrightarrow true\ U\ \Psi$

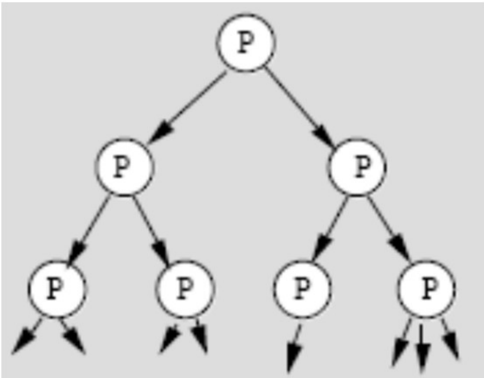
EXAMPLES



On each path there exists a state in which P holds true
AF P (= ¬E¬F P)



There exists an infinite path on which P holds in each state
EG P (= E ¬F ¬P)

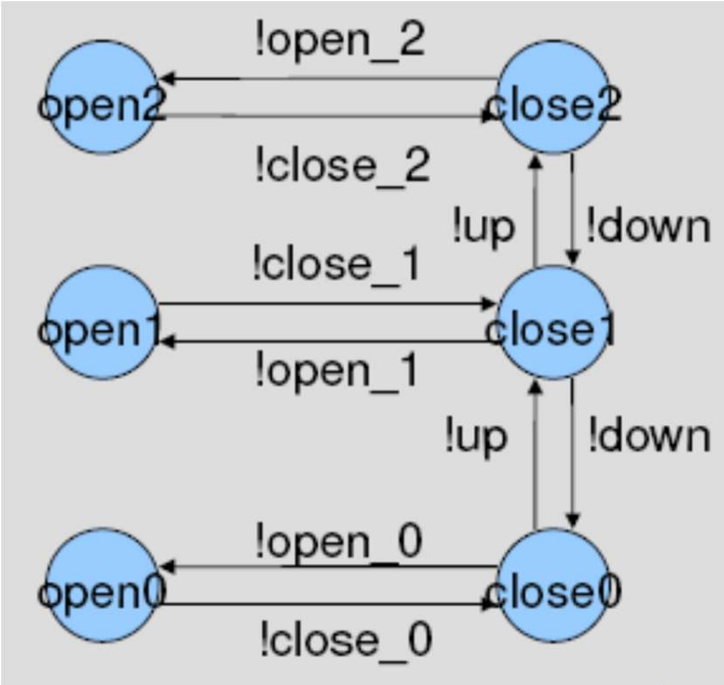


In all reachable states, P holds true
AG P (= ¬EF ¬P)

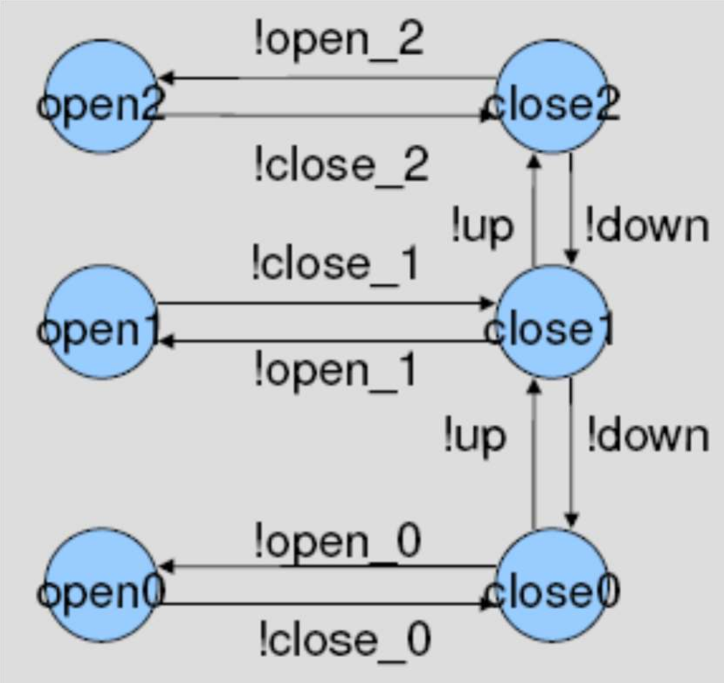
The temporal operators are of two types:
- on an execution (a path) – E
- on all executions (all paths) – A

Models - Reminder

- A model is:
  - A finite set of states, S
  - Some initial state s_0
  - A transition relation between states, $T \subseteq S \times S$
  - A finite set of atomic propositions, AP
  - A labelling function $L : S \rightarrow P(AP)$

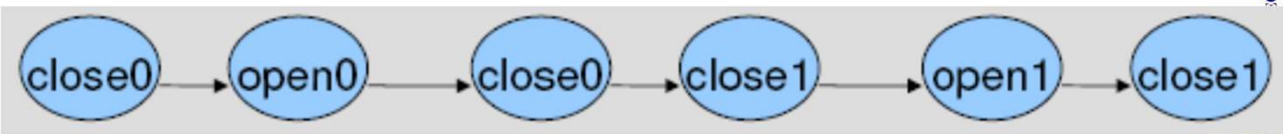


Formulas associated to the states of the automaton
L(openi) = {open, level = i}, i=0,1,2
L(closei) = {¬open , level = i } i=0,1,2



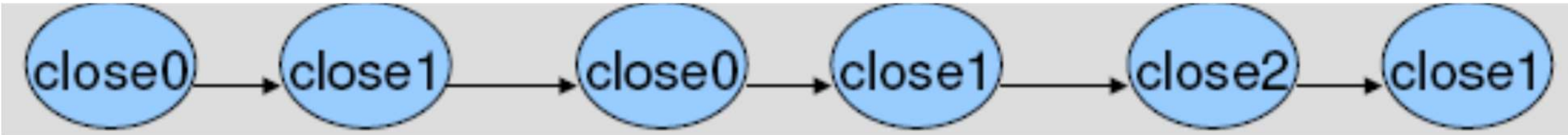
Formulas associated to the states of the automaton
L(openi) = {open, level = i}, i=0,1,2
L(closei) = {¬open , level = i } i=0,1,2

an execution of the automaton

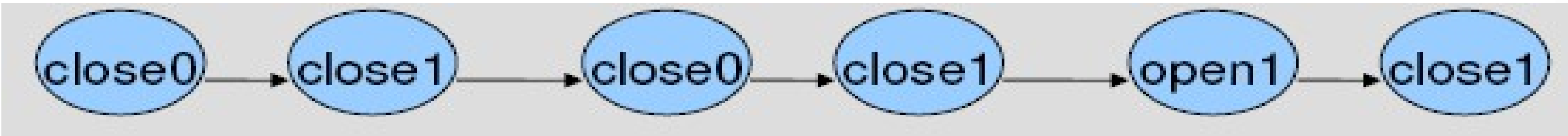


s,0 |= X open   s,0 |= F ¬close
s,2 |= X ¬open ∧ X level = 1
s,i |= G F ¬open   i = 0,…, 5

Notation: s|= P $\Leftrightarrow$ s,0 |= P



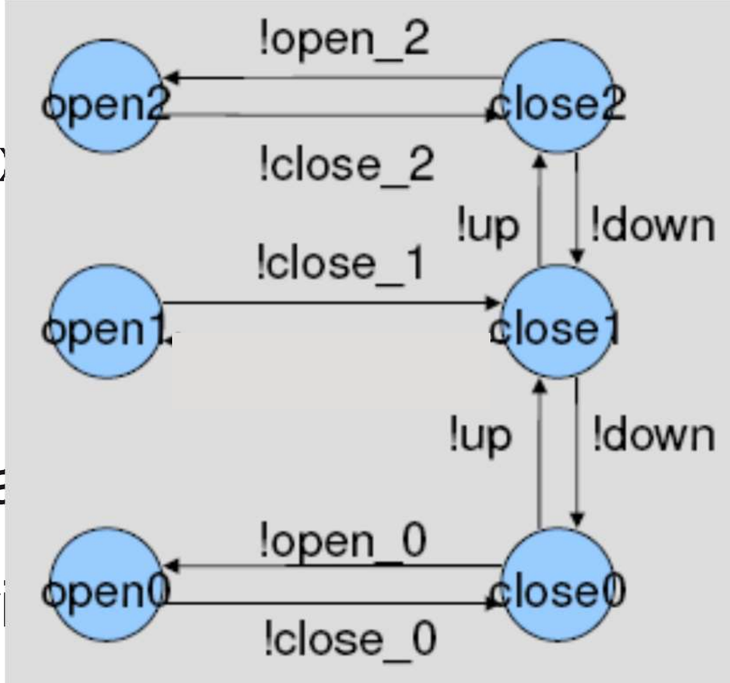
s |= G ¬open



s |= ¬open U level = 1

Temporal operators on all executions : A, E

- A φ : all the executions starting from the current state satisfy φ
- E φ : there e[...] from the current state φ
- E F φ : we ca[...] *afety property*
- A F φ : we wi[...] *eness property*



s,3 |= A X ¬open

Examples

Asc the controller of the lift :

Asc |= E G ¬open

Asc |= AG (open ⇒ AX ¬open)

Asc |= AG (¬open ⇒ EX open)

Precise definition of CTL

- Syntactical restrictions:
  - Each temporal operator X, F, G, U have to be on immediate scope of a A or E , the combinations are:
    - **AX,AF, AG, AU, EX, EF, EG, EU**

- Syntax: atomic propositions are CTL formulas
  - if f and g are CTL formulas, then

  $\neg$f, f $\wedge$ g, AX f, EX f, A(fUg), E(fUg) are also CTL formulas

- Extensions :
  - f $\vee$ g = $\neg$($\neg$f $\wedge$ $\neg$g)
  - AF g = A(true U g) EF g = E(true U g)
  - AG f = $\neg$E(true U $\neg$f) EG f = $\neg$A(true U $\neg$f)

Semantic of CTL

- s |= f (f atomic) **iff** f $\in$ L(s)
- s |= ¬f **iff** s |≠ f
- s |= f $\wedge$ g **iff** s|=f and s|=g
- s,0 |= AX f **iff** for all s such that s_0 = s,0, s,1|= f
- s,0 |= EX f **iff** it exists a s such that s_0 = s,0 and s,1|= f
- s,0 |= A (f U g) **iff** for all s s.t. s_0 = s,0, it exists i≥0 s.t. s,i |= g and for all j<i, s,j |= f
- s,0 |= E (f U g) **iff**

  it exists a s s.t. s_0 = s,0 and

  it exists i≥0 s.t. s,i |= g and

  for all j<i, s,j |= f

Cons and pro of CTL

☺ Model checking of linear complexity

☹ difficulties or unwillingness to express some kinds of properties (but they are advanced techniques resolving that issue!)

Other temporal logics:

CTL*, PLTL (PSPACE complet), FCTL (*Fairness*), TCTL (*Timers*), Logics with *past:* no model-checkers.

Exercises

- **See the PDF**