Examples:
Modelling formalisms, temporal logics, model checking

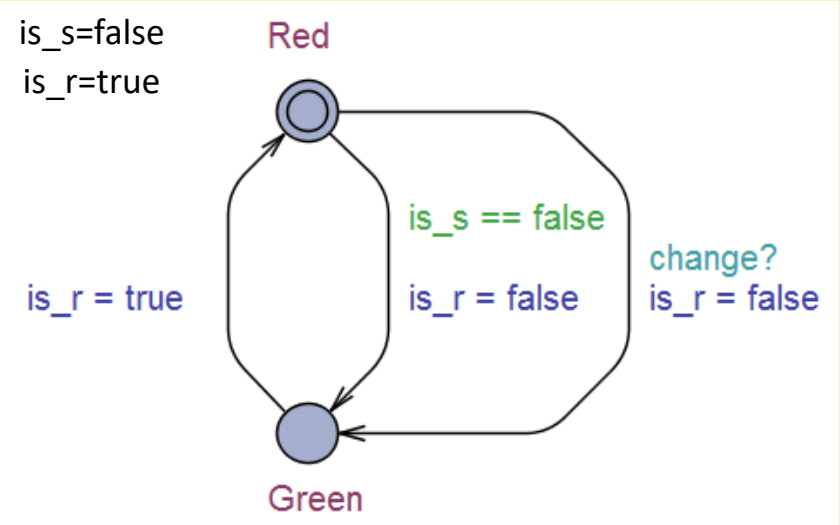
dr. István Majzik
BME Department of Measurement and Information Systems

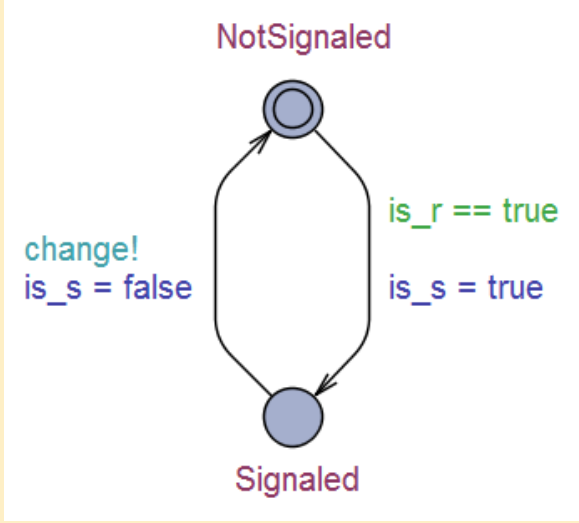
Interpretation of formal models

Interpretation of automata models

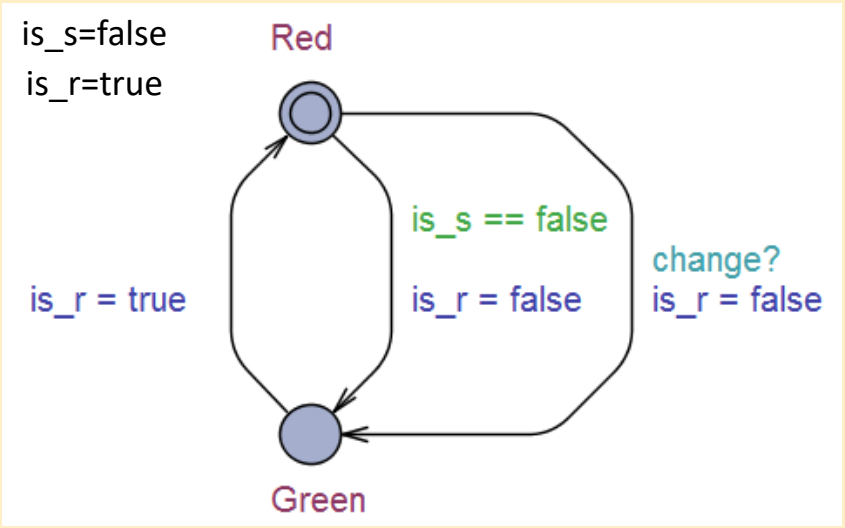
Consider the following two automata models (constructed in the UPPAAL tool) that model the behavior of a traffic light and a pedestrian. In the initial state is_r=true, is_s=false.

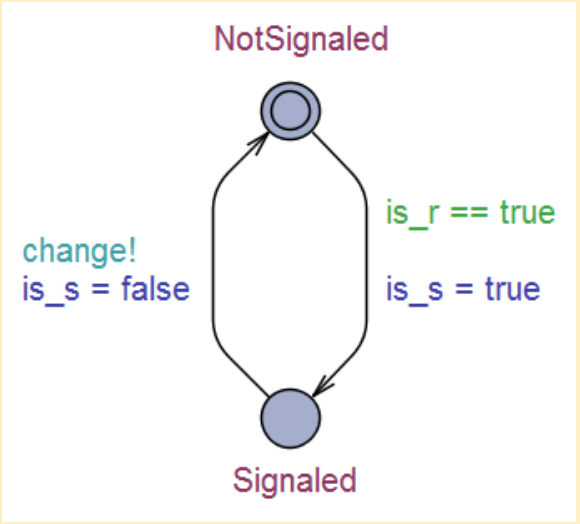
- Draw the Kripke structure corresponding to the whole system, i.e., the reachable combinations of the states of the traffic light and the pedestrian, and the related transitions!
  Label each combined state with the names of the states that it represents (you can use the initial letters of the states).

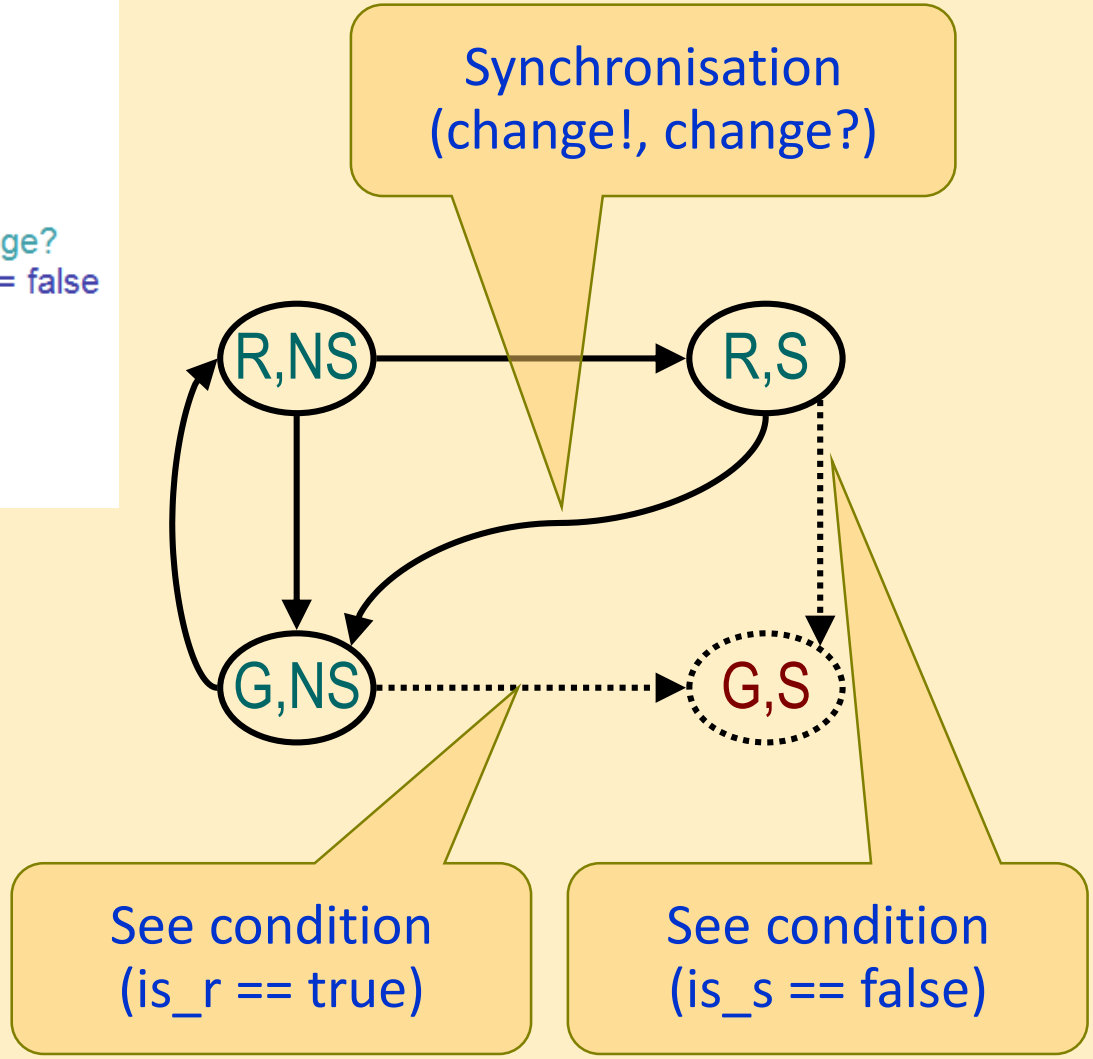




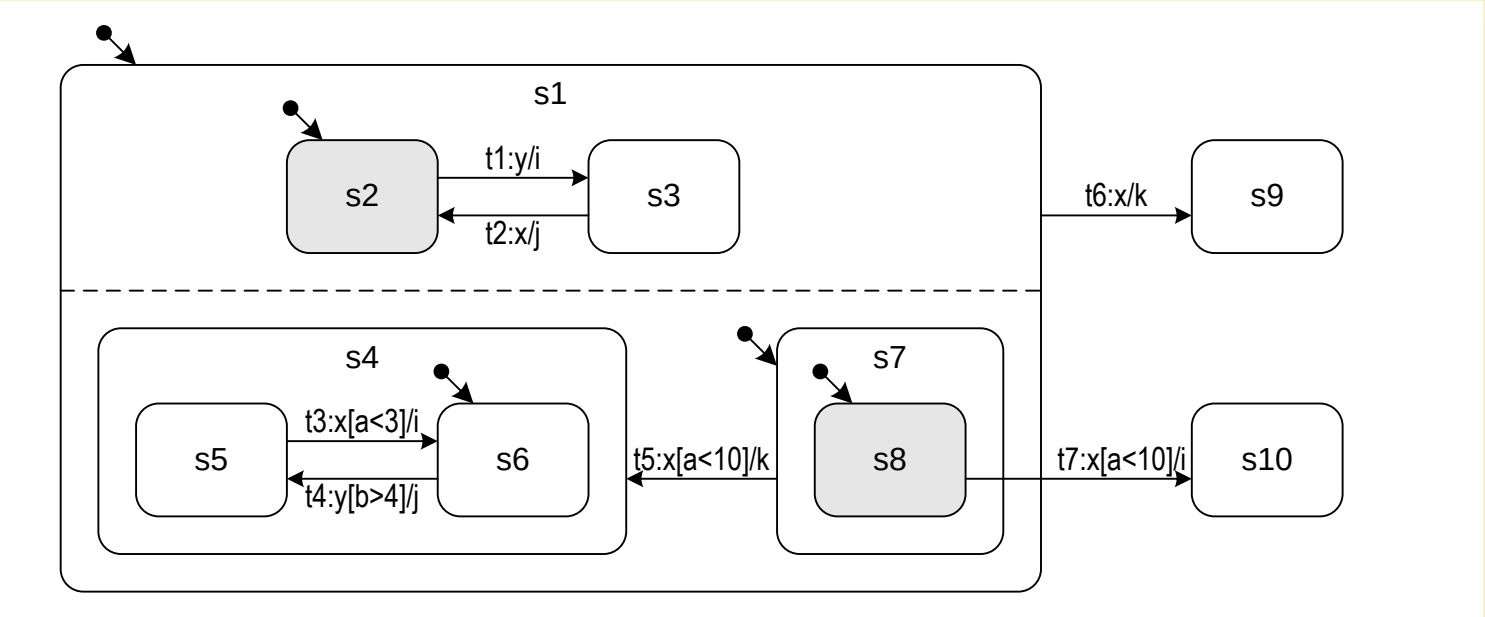
Interpretation of automata models







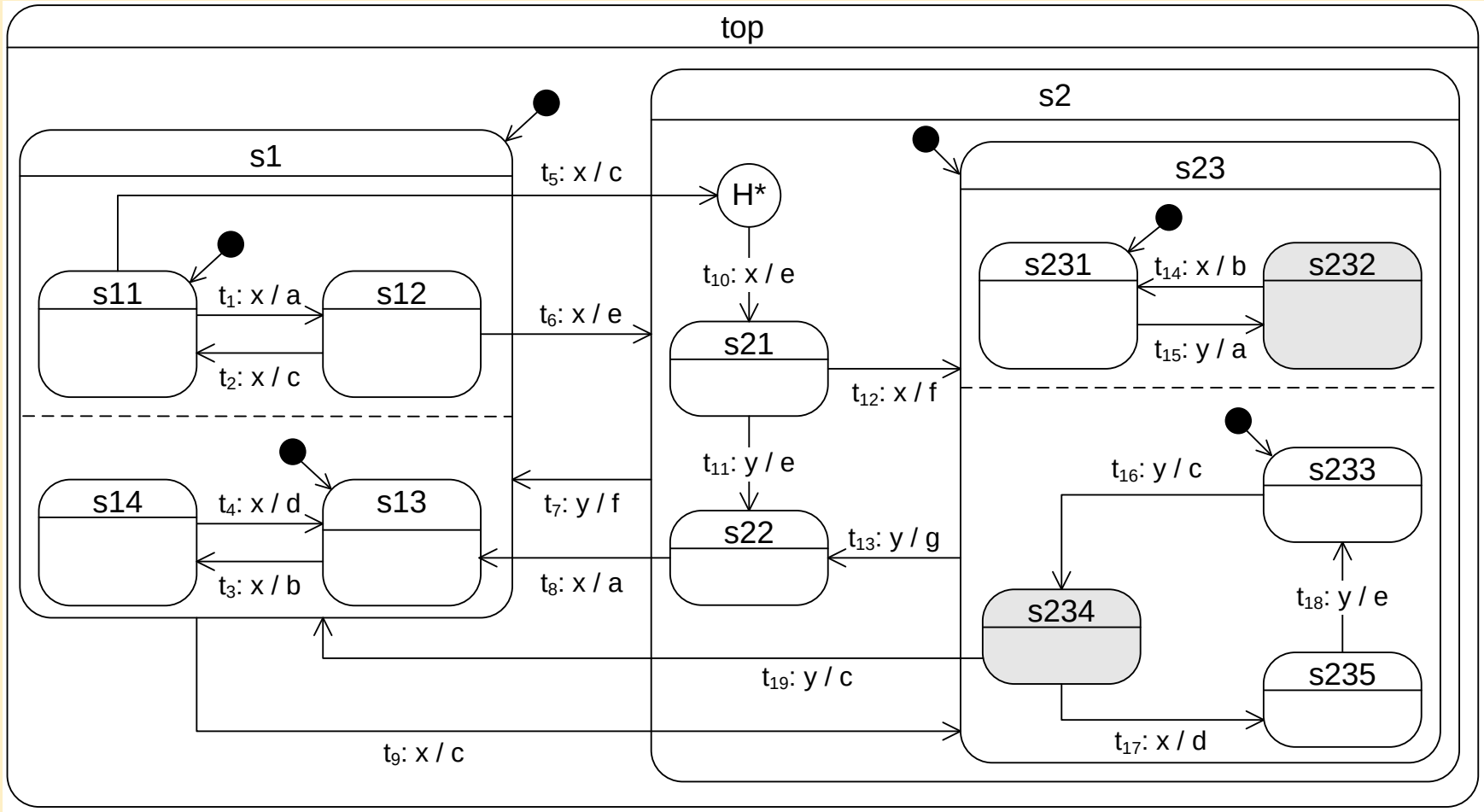
Interpretation of statecharts (1)



In the initial state configuration, the value of variable *a* is 8 and an event "*x*" occurs.

1. Which transitions are enabled?
2. Which enabled transitions are in conflict?
3. What is the set of fireable transitions?
4. What is (are) the next stable state configuration(s)?
5. What actions are executed and in what order?

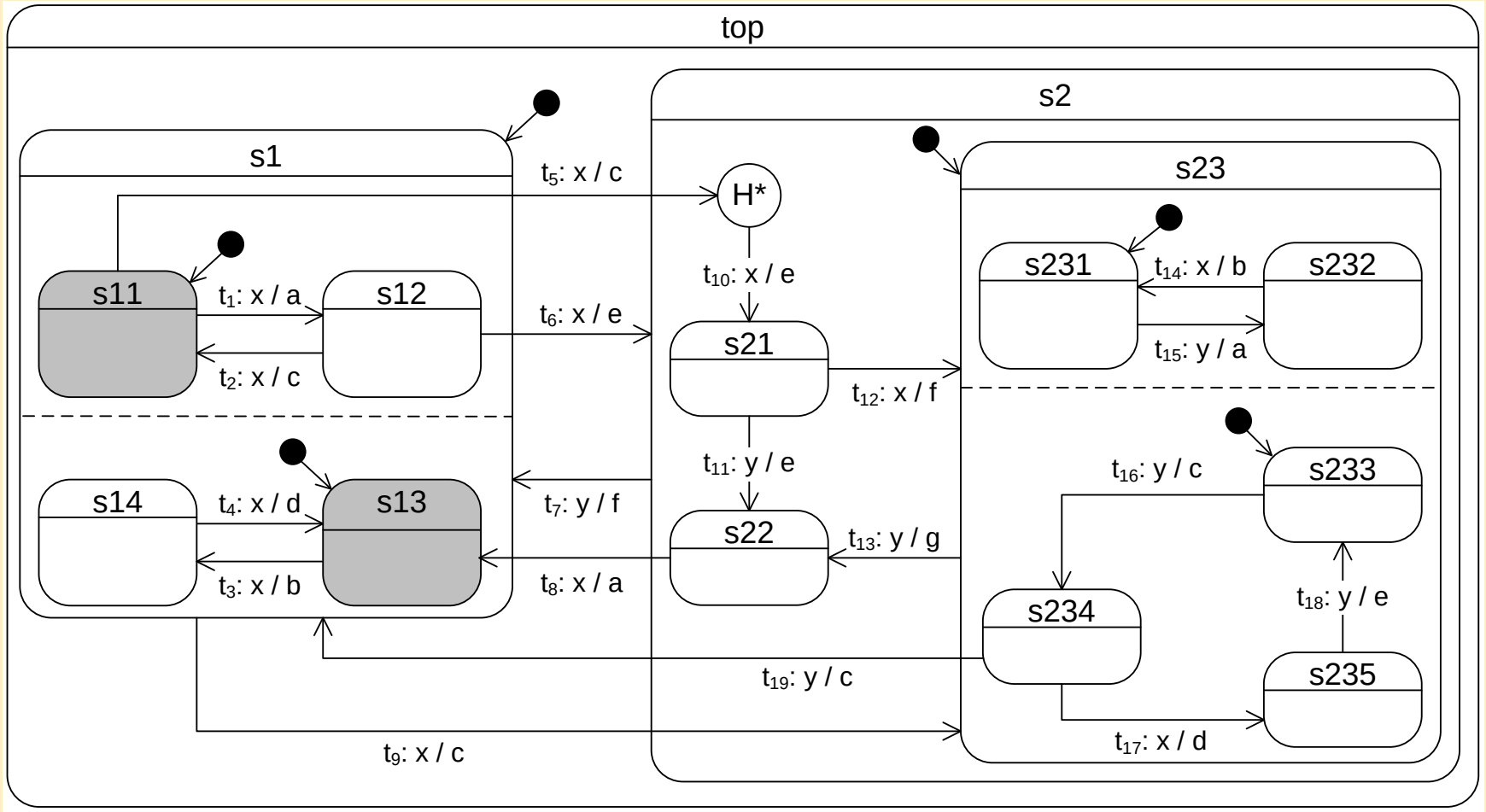
Interpretation of statecharts (2.1)



In the state configuration {top, s2, s23, s232, s234} the event *y* is passed by the scheduler.

- What will be the new state configuration?

Interpretation of statecharts (2.2)



After this, the event *x* is passed by the scheduler.

1. Which transitions are enabled, in conflict and fireable?
2. What will be the new state configuration? What actions are executed?

Formalization of properties using temporal logics

Theoretical questions

Argue if the following LTL equivalences are correct or not:

1. (F Stop) V (F Start) ≡ F (Stop V Start)
2. G Stop ≡ not F (not Stop)

Argue if the following CTL equivalences are correct or not:

1. AF (Start $\vee$ Stop) ≡ (AF Start) $\vee$ (AF Stop)
2. AF (Start $\wedge$ Stop) ≡ (AF Start) $\wedge$ (AF Stop)
3. EF (Start $\wedge$ Stop) ≡ (EF Start) $\wedge$ (EF Stop)

Argue if the following formula are syntactically correct in CTL or not!

1. A (X Stop $\vee$ F Start)
2. A (Stop U (AX Start))

Requirement formalization: Railway crossing

- We model the behavior of a railway crossing signal with the following atomic labels:

  {off, white, red}

- The behavior of the driver arriving at the crossing is modeled with the following atomic labels:

  {arriving, looking, stopping, crossing}

- Use LTL expressions to formalize the following properties which apply to the behavior of the driver in every case:

  1. If the signal is off, the driver will be looking and then in the next moment either stopping or crossing.
  2. The driver will eventually cross the crossing.
  3. If upon arrival, the signal is red, the driver will not cross until the signal is white.

Requirement formalization: Server room

- We model the states of a server performing complex simulation with the following atomic labels:
  {off, waiting, warm-up, simulation}
- The air-conditioning system is modeled with the following atomic labels:
  {stand-by, normal, maximal}
- Use LTL expressions to formalize the following properties which apply to the behavior of the server in every case:
  1. If in any moment the simulation is performed with the air-conditioning system being in the stand-by state, then in the next moment, the server will move to the waiting state.
  2. Eventually, the simulation will be started.
  3. The simulation can be performed only if there has been a warm-up phase with the air-conditioning system in the normal state.

Model checking algorithms

Theoretical basis for the algorithms

1. Draw the rule for tableau construction in case of the operator U of PLTL.
   Describe in which cases will the tableau branch be contradicting in case of a formula P U Q!
2. Describe how to identify the states (of a Kripke structure) in which the formula A(P U Q) holds!
3. Describe the necessary steps to construct an ROBDD from a binary decision tree!
4. Describe the basic idea of bounded model checking!

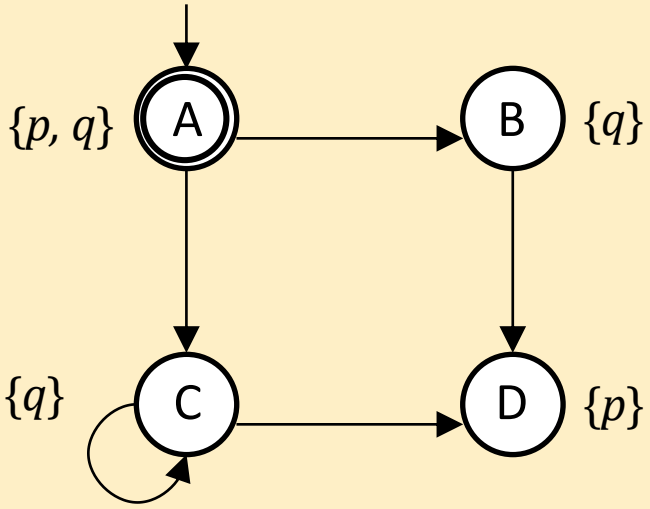
Checking CTL using iterative labeling

Consider the Kripke structure given below.

- Check if the following CTL expression holds from the initial state using the iterative labeling algorithm presented in the lectures:

$$\mathbf{A}(p\ \mathbf{U}\ (\mathbf{EX}\ \neg q))$$

For each iteration give the expression that is currently used for labeling and enumerate the states that are labeled!

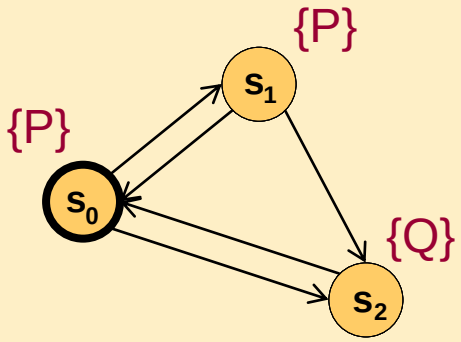


Model checking with the tableau method

Consider the Kripke structure on the right.

Perform the model checking of the following formula with the tableau method:

$$\neg (P \; U \; Q)$$

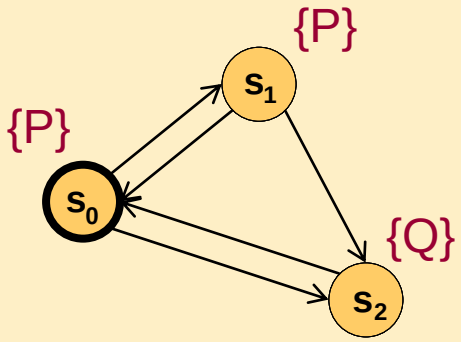


Model checking with the tableau method

Consider the Kripke structure on the right.

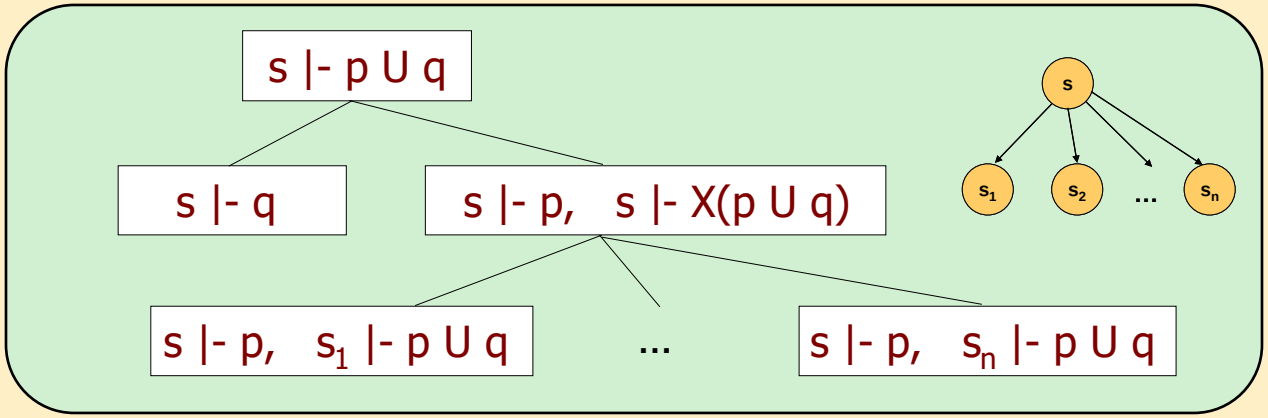
Perform the model checking of the following formula with the tableau method:

$\neg$ (P U Q)



Things to know:

- Negation (to look for counterexamples): (P U Q)
- Tableau rule: (P U Q) = Q $\vee$ (P $\wedge$ X(P U Q))

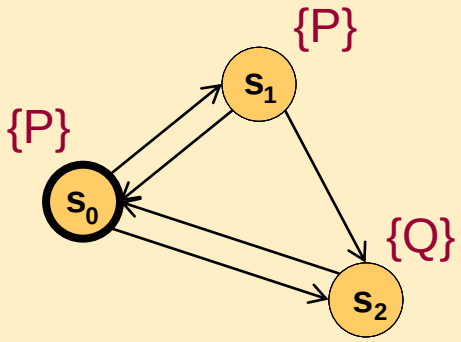


Model checking with the tableau method

Consider the Kripke structure on the right.

Perform the model checking of the following formula with the tableau method:

$$\neg (P \; U \; Q)$$

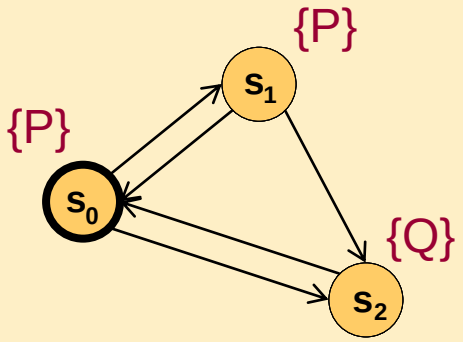


Things to know:

- Negation (to look for counterexamples): $(P \; U \; Q)$
- Tableau rule: $(P \; U \; Q) = Q \vee (P \wedge X(P \; U \; Q))$
- Contradicting branch if:
  - Atomic proposition does not hold in a state
  - X operator with deadlock
  - Loop with P, but without Q
- Satisfying branch (here: giving counterexamples) if:
  - Only atomic propositions and all of them hold in the state
  - Cycle without contradiction

Model checking with the tableau method

Tableau construction:



s_0 |- P U Q

s_0 |- Q

Contradicting branch

s_0 |- P, s_0 |- X(P U Q)

s_1 |- P U Q

s_2 |- P U Q

s_1 |- Q

Contradicting branch

s_1 |- P, s_1 |- X(P U Q)

s_2 |- Q

Satisfying branch

s_2 |- P, s_2 |- X(P U Q)

Contradicting branch

s_0 |- P U Q

Contradicting branch: Loop with P only

s_2 |- P U Q

Existing tableau node

Counterexample for ¬(P U Q):
s_0, s_2
s_0, s_1, s_2

Model checking: Servers

- An IT system has two servers, a database server and an application server, both of which can be turned on or off.
- Initially, both servers are off. During normal operation, the servers are turned on and off simultaneously.
- The system is functional if both servers are on.
- If – in the functional state – the database server is turned off due to an error, the system becomes nonfunctional. After this, the application server is also turned off, then the system is restarted by turning both servers on again.
- Tasks:
  1. Create a Kripke structure modeling the behavior of the system described above with regard to the states of the servers! Label the states with the following atomic propositions (based on the informal description):
     {initial, functional, nonfunctional}
  2. Check if the following CTL formula holds for the functional state of the Kripke structure:
     E(¬nonfunctional U initial)

Model checking: Behavior of a student

- The "states" of a student are characterized by two predicates: drinking coffee or not, and sleeping or not.
- The student has three activities:
  - During studying, she is drinking coffee and not sleeping;
  - After this she is taking an exam, when she is not drinking coffee and not sleeping either;
  - After exam, she is resting, when she is sleeping and not drinking coffee.
- The initial activity of the student is studying, which is continuous until taking an exam. She will not take an exam without studying and will study only after resting.
- Tasks:
  1. Create a Kripke structure modeling the behavior of the student described above with regard to drinking coffee and/or sleeping! Label the states with the following atomic propositions (based on the informal description): {resting, studying, taking_exam}
  2. Check if the following CTL formula holds on the model (the initial state is studying, as specified in the text):

$$E(\neg taking\_exam\ U\ resting)$$

ROBDD: Building ROBDD

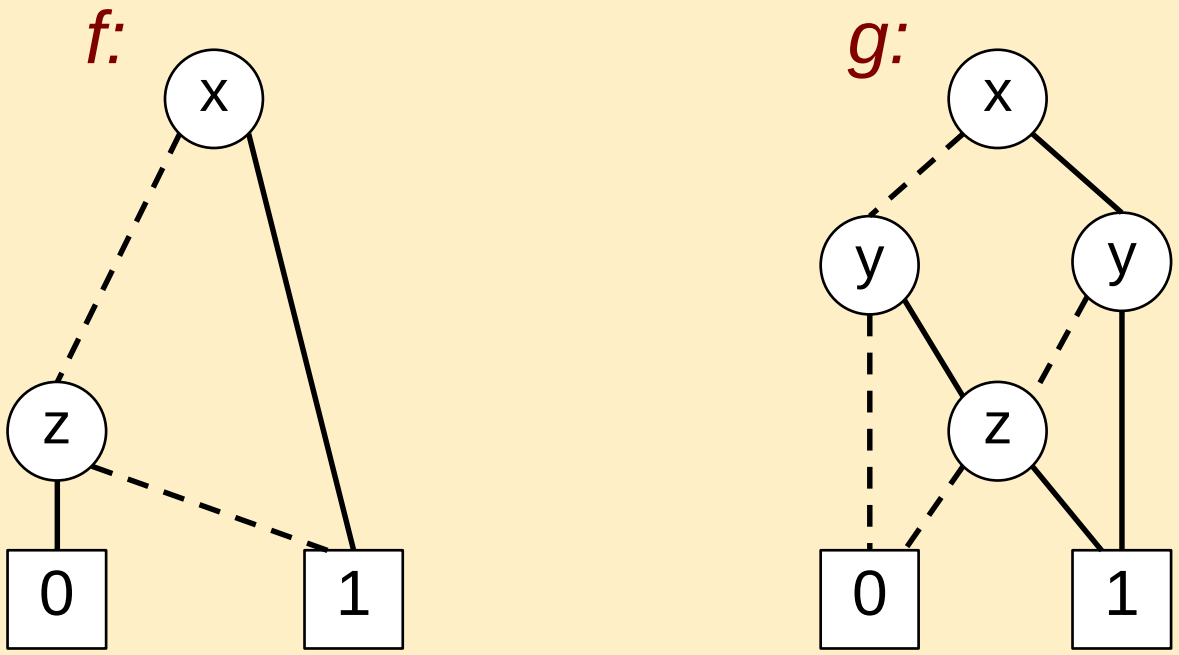
Consider the following Boolean function *g*:

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

1. Construct the decision tree representing *g*! Use the variable ordering used in the table: *x*, *y*, *z*.
2. Based on this, construct the reduced ordered binary decision diagram (ROBDD) representation of *g*!
3. Give the algebraic form of the function!

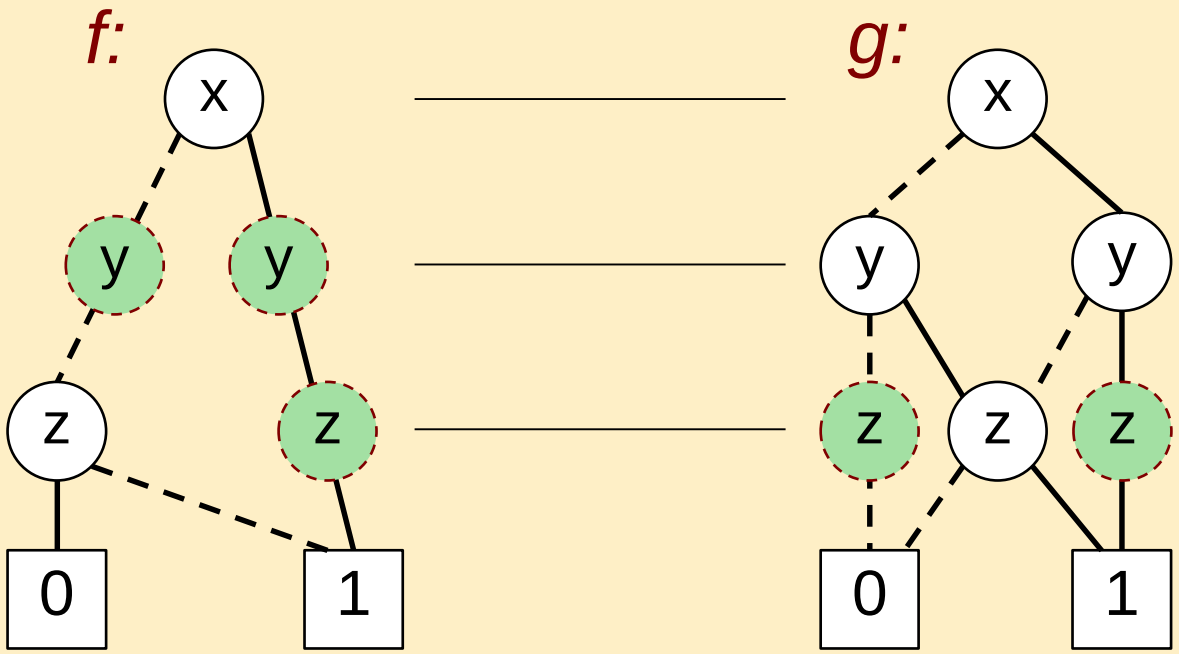
ROBDD: Operations on functions

Consider the following functions f and g given in ROBDD form. Construct the ROBDD representing $f \wedge g$!



ROBDD: Operations on functions

Consider the following functions f and g given in ROBDD form. Construct the ROBDD representing $f \wedge g$!



ROBDD: Operations on functions

Consider the following functions f and g given in ROBDD form. Construct the ROBDD representing $f \wedge g$!