AGH
University of Science and Technology in Kraków
Faculty of Electrical Engineering, Automatics, Computer Science and Biomedical Engineering
Department of Applied Computer Science

# Model checking – laboratory classes

**Marcin Szpyrka**

Kraków 2017

# Contents

# Chapter 1

# Introduction

The script contains a set of exercises for the *Model checking* course. Some of the exercises will be solved during laboratory classes. The rest are intended for homework as preparation for the exam.

The knowledge needed to solve the exercises is presented during lectures. Necessary information can also be found on slides for lectures and in recommended literature. The main textbook for the course is the Principles of Model Checking by Baier and Katoen [1].

**The last chapter contains a sample exam!**

Information about the errors found in the script, please send to the email address:
`mszpyrka@agh.edu.pl`

# Chapter 2

# Transition systems

**Exercise 2.1.** Automatic Train Protection (ATP) systems are used to guarantee a train safety even if the driver is not capable of controlling the train. In the ATS system, a light signal is turned on every 60 seconds to check whether the driver controls the train. If the driver fails to acknowledge the signal within 6 seconds, a sound signal is turned on. Then, if the driver does not disactivate the signals within 3 seconds, using the acknowledge button, the emergency brakes are applied automatically to stop the train.

Start **nuXmv** in interactive mode. Read and compile the **ATP** system model. A few times run the simulation in the interactive mode (at least 70 steps) to test possible ways of the model behaviour.

Listing 2.1: ATP model

```
MODULE main
VAR
  light : boolean;
  sound : boolean;
  brake : boolean;
  button : boolean;
  delay : 0 .. 9;
  timer : 0 .. 60;

ASSIGN
  init(light) := TRUE;
  init(sound) := FALSE;
  init(brake) := FALSE;
  init(button) := FALSE;
  init(delay) := 0;
  init(timer) := 0;

  next(timer) := case
    timer = 59 : 0;
    timer < 60 : timer + 1;
    TRUE : timer;
  esac;

  next(delay) := case
    button = TRUE : 0;
    button = FALSE & light = TRUE & delay < 9 : delay + 1;
    TRUE : delay;
  esac;

  next(light) := case
    timer = 59 & light = FALSE : TRUE;
    light = TRUE & button = TRUE : FALSE;
    TRUE : light;
```

```
  esac;

  next(sound) := case
    delay = 6 & button = FALSE : TRUE;
    sound = TRUE & button = TRUE : FALSE;
    TRUE : sound;
  esac;

  next(brake) := case
    delay = 9 & button = FALSE : TRUE;
    TRUE : brake;
  esac;

  next(button) := case
    button = TRUE : FALSE;
    light = TRUE & brake = FALSE : {FALSE, TRUE};
    TRUE : FALSE;
  esac;
```

**Exercise 2.2.** Reset **nuXmv** environment. Read and compile the ATM model. A few times run the simulation in the interactive mode (at least 10 steps) to test possible ways of the model behaviour.

Listing 2.2: ATM model

```
MODULE main
VAR
  s: {welcome, enterPin1, enterPin2, enterPin3, tryAgainPin2, tryAgainPin3,
  cardTaken, askAmount1, askAmount2, askAmount3, tryAgainAmount2,
  tryAgainAmount3, takeMoney, takeCard, thanksGoodbye, sorry};

IVAR
  a: {cardIn, correctPin, wrongPin, ack, cancel, fundsOK, fundsWrong,
  moneyOut, cardOut, none};

ASSIGN
  init(s) := welcome;

  next(s) := case
    s = welcome         & a = cardIn      : enterPin1;
    s = enterPin1       & a = correctPin  : askAmount1;
    s = enterPin1       & a = wrongPin    : tryAgainPin2;
    s = enterPin1       & a = cancel      : takeCard;
    s = enterPin2       & a = correctPin  : askAmount1;
    s = enterPin2       & a = wrongPin    : tryAgainPin3;
    s = enterPin2       & a = cancel      : takeCard;
    s = enterPin3       & a = correctPin  : askAmount1;
    s = enterPin3       & a = wrongPin    : cardTaken;
    s = enterPin3       & a = cancel      : takeCard;
    s = tryAgainPin2    & a = ack         : enterPin2;
    s = tryAgainPin2    & a = cancel      : takeCard;
    s = tryAgainPin3    & a = ack         : enterPin3;
    s = tryAgainPin3    & a = cancel      : takeCard;
    s = cardTaken                         : sorry;
    s = askAmount1      & a = fundsOK     : takeMoney;
    s = askAmount1      & a = fundsWrong  : tryAgainAmount2;
    s = askAmount1      & a = cancel      : takeCard;
    s = askAmount2      & a = fundsOK     : takeMoney;
    s = askAmount2      & a = fundsWrong  : tryAgainAmount3;
```

```
    s = askAmount2       & a = cancel      : takeCard;
    s = askAmount3       & a = fundsOK     : takeMoney;
    s = askAmount3       & a = fundsWrong  : takeCard;
    s = askAmount3       & a = cancel      : takeCard;
    s = tryAgainAmount2  & a = ack         : askAmount2;
    s = tryAgainAmount2  & a = cancel      : takeCard;
    s = tryAgainAmount3  & a = ack         : askAmount3;
    s = tryAgainAmount3  & a = cancel      : takeCard;
    s = takeMoney        & a = moneyOut    : takeCard;
    s = takeMoney        & a = none        : takeCard;
    s = takeCard         & a = cardOut     : thanksGoodbye;
    s = thanksGoodbye                      : welcome;
    s = sorry                              : welcome;
    TRUE                                   : s;
  esac;

TRANS  s = welcome         ->
   (a = none | a = cardIn)
TRANS  s = enterPin1       ->
   (a = none | a = correctPin | a = wrongPin | a = cancel)
TRANS  s = enterPin2       ->
   (a = none | a = correctPin | a = wrongPin | a = cancel)
TRANS  s = enterPin3       ->
   (a = none | a = correctPin | a = wrongPin | a = cancel)
TRANS  s = tryAgainPin2    ->
   (a = none | a = ack | a = cancel)
TRANS  s = tryAgainPin3    ->
   (a = none | a = ack | a = cancel)
TRANS  s = askAmount1      ->
   (a = none | a = fundsOK | a = fundsWrong | a = cancel)
TRANS  s = askAmount2      ->
   (a = none | a = fundsOK | a = fundsWrong | a = cancel)
TRANS  s = askAmount3      ->
   (a = none | a = fundsOK | a = fundsWrong | a = cancel)
TRANS  s = tryAgainAmount2 ->
   (a = none | a = ack | a = cancel)
TRANS  s = tryAgainAmount3 ->
   (a = none | a = ack | a = cancel)
TRANS  s = takeMoney       ->
   (a = none | a = moneyOut)
TRANS  s = takeCard        ->
   (a = none | a = cardOut)
TRANS  s = cardTaken       ->
   (a = none)
TRANS  s = thanksGoodbye   ->
   (a = none)
TRANS  s = sorry           ->
   (a = none)
```

**States**:

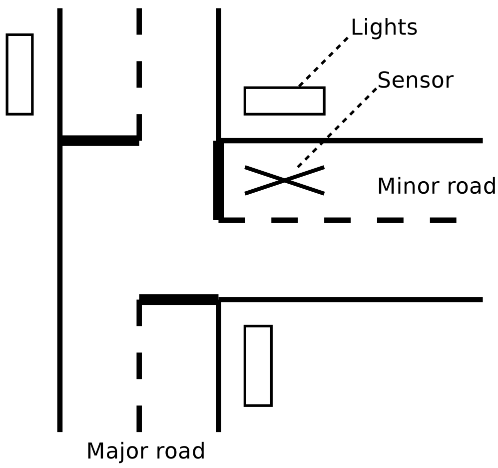
- *welcome* – the machine is idle;
- *enterPin1* – the first attempt to enter a valid PIN, similarly *enterPin2* and *enterPin3*
- *tryAgainPin2* – after entering invalid PIN the machine asks whether the user wants to try again, similarly *tryAgainPin3*

- *cardTaken* – the ATM card is taken after three unsuccessful attempts to enter the valid PIN
- *askAmount1* – the first attempt to enter a valid amount to take, *valid* means here that there is enough money in the bank account, similarly *askAmount2* and *askAmount3*
- *tryAgainAmount2* – after entering invalid amount the machine asks whether the user wants to try again, similarly *tryAgainAmount3*
- *takeMoney* – the ATM pays the money
- *takeCard* – the ATM returns the ATM card
- *thanksGoodbye* – the ATM displays a message *thank you, good bye*
- *sorry* – the ATM displays a message *sorry, the operation can not be performed*

**Actions**:

- *cardIn* – the user inserts the ATM card into the ATM
- *correctPin* – the user enters the valid PIN
- *wrongPin* – the user enters an invalid PIN
- *ack* – the user confirms that he wants to try again
- *cancel* – the user resigns from the next attempt
- *fundsOK* – the user enters a valid amount
- *fundsWrong* – the user enters an invalid amount
- *moneyOut* – the user takes the money
- *cardOut* – the user takes the ATM card
- *none* – an empty action, it means *nothing happens*

**Exercise 2.3.** Develop a **nuXmv** model for the following traffic lights system. Use the simulation to preliminary check the model correctness.

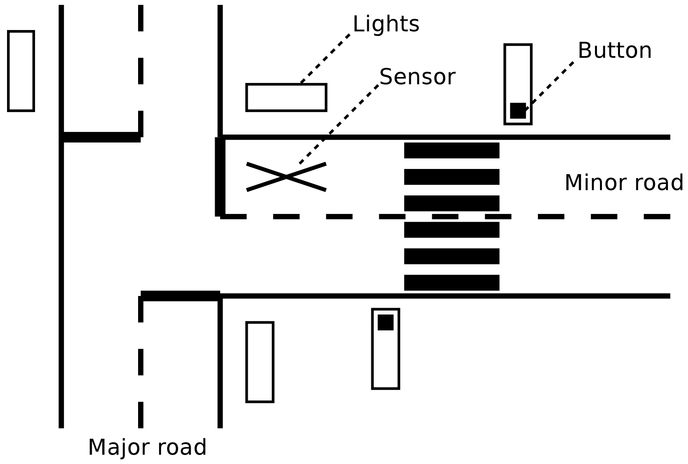



A control system must ensure safe and correct functioning of a set of traffic lights at a T-junction between a major and a minor road. The lights will be set to green on the major road and red on the minor road unless a vehicle is detected by a sensor in the road just before the lights in the minor road. In this case the lights will be switchable in the standard manner and allow traffic to leave the minor road. After a suitable interval the lights will revert to their default position to allow traffic to flow on the major road again. Once a vahicle is detected the sensor will be disable until the minor-road lights are set to red again.

The exercise is based on the example presented in [2]. The book contains CCS model for this problem.

**Exercise 2.4.** Expand the previous model, as described below.

There is also to be a pedestrian crossing a short distance down to minor road but beyond the sensor. There is a button on each side of the road for pedestrians to indicate they wish to cross. The crossing should only allow people to cross when the 'minor lights' are set to red in order to minimise waiting times for traffic on the minor road. All requests for service from either the sensor or the button must eventually be complied with ([2]).




**Exercise 2.5.** Develop a **nuXmv** model for the following home heating system. Use the simulation to preliminary check the model correctness.

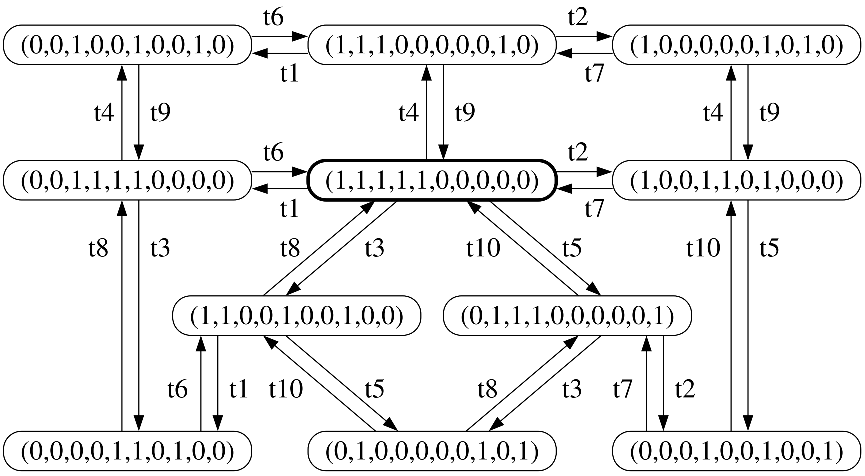
A temperature sensing device compares the difference between $Ta$, the temperature sensed in the house, and the reference temperature $Tr$, which is the desire house temperature. The difference between these two, the error in the temperature, is measured and sent to the controller. Users may set the reference temperature within the range 0..35 degrees Celsius. The controller will attempt to keep the house temperature within 2 degrees either way of the reference temperature by turning a furnace on or off as the allowable limits of the range are reached. To achieve this the controller sends discrete signals to start an ignition system and then a motor and shut them down in the reverse order. The furnace sends discrete signals to the controller to indicate the current state of motor rpm and ignition. If the motor does not reach its optimum speed on startup the furnace will signal an error which should abort the startup sequence. The controller will wait 5 seconds for a successful motor status signal from the furnace. It will terminate the ignition sequence if the unsuccessful signal is received or if no signal arrives within 5 seconds. On receiving an ignition error signal from the furnace, the controller will shut the system down and set an appropriate light on an abnormal status panel. If an error indicator is set on, a manual reset is required before the system may restart. There is a master switch which may be set on or off by users. A minimum of 5 minutes elapse between turning off the furnace and restarting it. The furnace must be shut down within 5 seconds if any furnace error is detected or the master switch is set to off ([2]).

**Remark:** To set the desired temperature we may use two buttons **up** and **down**, which respectively increase and reduce the temperature by 1 degree.

**Exercise 2.6.** Develop a **nuXmv** model for the given reachability graph (**see slides 30–31, part 1**).

Take into account the following atomic propositions:

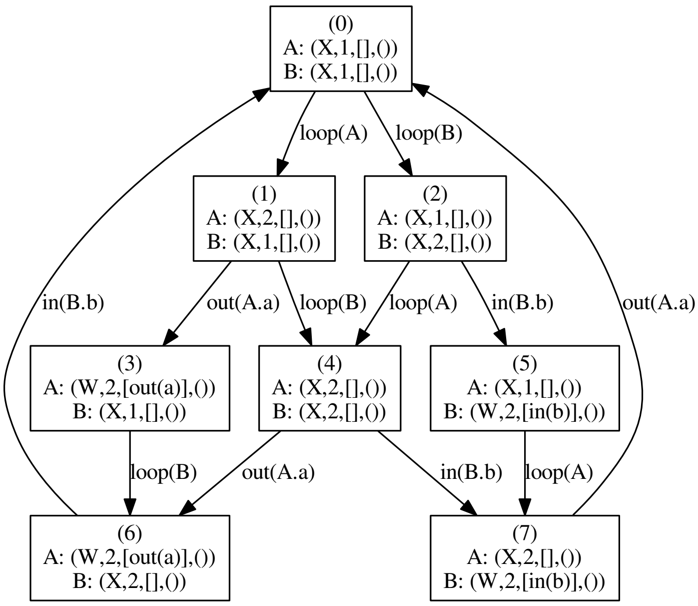
- $AP = \{w_1, \ldots, w_5, e_1, \ldots, e_5\}$ ($w$ – waiting, $e$ – eating)




- $e_i \in L(M)$ iff $M(p(i+5)) = 1$
- $w_i \in L(M)$ iff $M(p(i+5)) = 0$

Use the simulation to preliminary check the model correctness.

**Exercise 2.7.** Develop a **nuXmv** model for the given LTS graph (**see slide 42, part 1**).




Define the $AP$ set so that you can analyse the agents' mode (possible values X and W) and program counter (possible values 1 and 2). Use the simulation to preliminary check the model correctness.

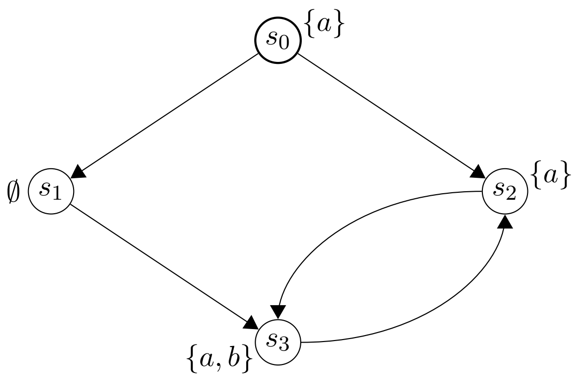
# Chapter 3

# Linear time properties

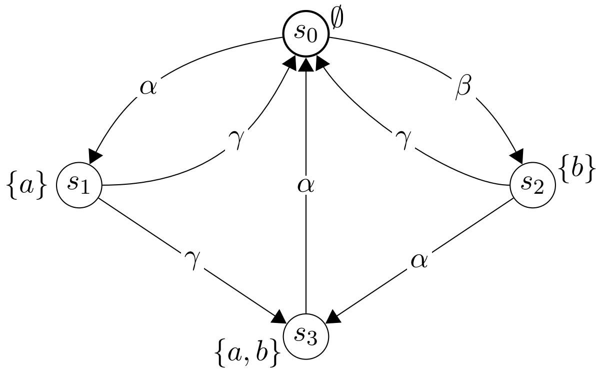
**Exercise 3.1.** Consider the set of atomic propositions defined by $AP = \{a, b, c\}$. Characterize each of the following linear-time properties as being either an invariant, safety property, liveness property, or none of these. Define the propositional logic formulas over $AP$ for invariants (**see slides 62, 66, part 1**).

a) $a$ should never occur.
b) $a$ should occur exactly once.
c) $a$ and $b$ alternate infinitely often.
d) $a$ should eventually be followed by $b$.
e) $a$ should occur at most 100 times.
f) $a$ should occur at least 100 times.
g) $b$ should occur at most as many times as $a$.
h) $b$ should occur at least as many times as $a$.
i) $a$ and $b$ should occur in total at least 10 times.
j) $a$ should occur every second state.
k) $a$, $b$, and $c$ never occur together in the same state.
l) $a$ and $b$ should occur together at most 100 times.
m) $c$ never occurs.
n) $a$, $b$, and $c$ always occur together.
o) If $a$ occurs in the given state then $b$ occurs in the next state.
p) $a$ never occurs in two consecutive states.

**Exercise 3.2.** Give the traces $Traces(TS)$ on the set of atomic propositions $AP = \{a, b\}$ of the following transition system $TS$ ([1]).
**Remark:** $I = \{s_0\}$.



**Exercise 3.3.** Consider the given transition system $TS$ and the sets of actions $B_1 = \{\alpha\}$, $B_2 = \{\alpha, \beta\}$ and $B_3 = \{\beta\}$ ([1]).



Let $E_1$, $E_2$, and $E_3$ be the following LT properties:

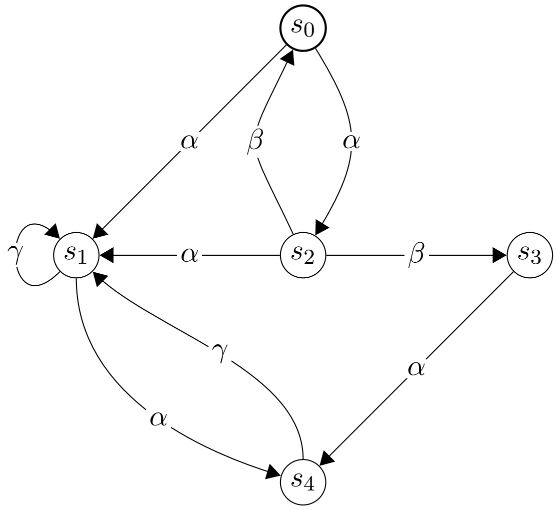
- $E_1$ is set of all words $A_0A_1 \ldots \in \left(2^{\{a,b\}}\right)^\omega$ with $A_i \in \{\{a,b\},\{b\}\}$ for infinitely many $i$.
- $E_2$ is set of all words $A_0A_1 \ldots \in \left(2^{\{a,b\}}\right)^\omega$ with $A_i \in \{\{a,b\},\{a\}\}$ for infinitely many $i$.
- $E_3$ is set of all words $A_0A_1 \ldots \in \left(2^{\{a,b\}}\right)^\omega$ for which there does not exist an $i \in \mathbb{N}$ such that $A_i = \{a\}$, $A_{i+1} = \{a,b\}$ and $A_{i+2} = \emptyset$.

**Questions:**

a) For which sets of actions $B_i$ ($i \in \{1,2,3\}$) and LT properties $E_j$ ($j \in \{1,2,3\}$) it holds that $TS \models_{\mathcal{F}_i} E_j$, where $\mathcal{F}_i = (\emptyset, \{B_i\}, \emptyset)$?

b) For which sets of actions $B_i$ ($i \in \{1,2,3\}$) and LT properties $E_j$ ($j \in \{1,2,3\}$) it holds that $TS \models_{\mathcal{F}_i} E_j$, where $\mathcal{F}_i = (\emptyset, \emptyset, \{B_i\})$?

**Exercise 3.4.** Consider the given transition system $TS$ ($AP = \emptyset$) ([1]).



Decide which of the following fairness assumptions $\mathcal{F}_i$ are realizable for $TS$. Justify your answers!

a) $\mathcal{F}_1 = (\{\{\alpha\}\}, \{\{\gamma\}\}, \{\{\alpha,\beta\}\})$;

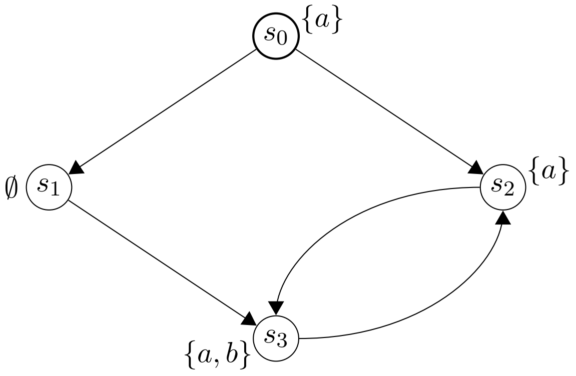
b) $\mathcal{F}_2 = (\{\{\alpha,\gamma\}\}, \{\{\alpha,\beta\}\}, \{\{\gamma\}\})$;

c) $\mathcal{F}_3 = (\{\{\alpha,\gamma\},\{\beta\}\}, \{\{\alpha,\beta\}\}, \{\{\gamma\}\})$;

# Chapter 4

# LTL logic

**Exercise 4.1.** Let the following transition system $TS$ be given ($I = \{s_0\}$).




Check (do not use **nuXmv**) whether $TS$ satisfies the following LTL formulas:

a) $\mathsf{GF}a$

b) $(\neg a \wedge \neg b) \Rightarrow \mathsf{XG}a$

c) $\mathsf{FXG}a$

d) $b \Rightarrow (\mathsf{G}a \wedge \mathsf{GF}b)$

e) $\mathsf{F}(\mathsf{G}b \vee \mathsf{G}a)$

Use the following SMV model and the **nuXmv** toolbox to check your answers.

Listing 4.1: SMV model

```
MODULE main
VAR
  s : {s0, s1, s2, s3};
  a : boolean;
  b : boolean;
ASSIGN
  init(s) := s0;

  next(s) := case
    s = s0 : {s1, s2};
    s = s1 : s3;
    s = s2 : s3;
    s = s3 : s2;
  esac;
```

```
a := case
  s = s1 : FALSE;
  TRUE   : TRUE;
esac;

b := case
  s = s3 : TRUE;
  TRUE   : FALSE;
esac;
```

**Exercise 4.2.** Assume $AP = \{a, b, c\}$. Define LTL formulas describing the following properties:

a) $a$ never occurs.

b) $a$ should occur exactly once.

c) $a$ should eventually be followed by $b$.

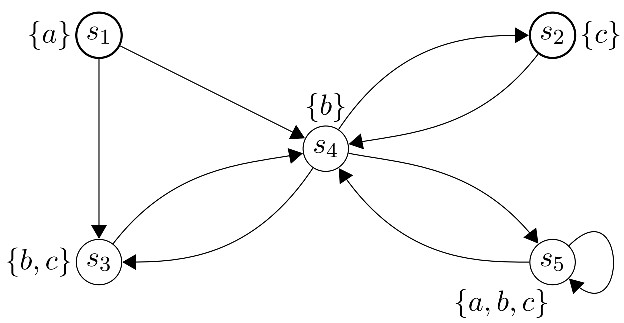
d) $a$, $b$, and $c$ never occur together in the same state.

e) $a$, $b$, and $c$ always occur together.

f) If $a$ occurs in the given state then $b$ occurs in the next state.

g) $a$ never occurs in two consecutive states.

**Exercise 4.3.** Let the following transition system $TS$ be given ($I = \{s_1, s_2\}$).




Check (do not use **nuXmv**) whether $TS$ satisfies the following LTL formulas:

a) $\mathsf{FG}c$

b) $\mathsf{GF}c$

c) $\mathsf{X}\neg c \Rightarrow \mathsf{XX}c$

d) $\mathsf{G}a$

e) $a \,\mathsf{U}\, \mathsf{G}(b \vee c)$

f) $(\mathsf{XX}b) \,\mathsf{U}\, (b \vee c)$

g) $\mathsf{FG}(a \vee c)$

h) $(a \Rightarrow \mathsf{X}(b \wedge ((b \vee c) \,\mathsf{U}\, a)))$

i) $(a \vee c) \wedge \mathsf{XG}(b \vee c)$

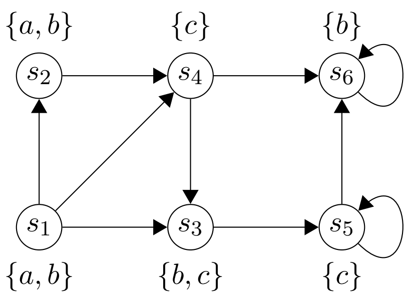
j) $\mathsf{XG}(\neg b \vee ((b \vee c) \,\mathsf{U}\, a))$

Use the following SMV model and the **nuXmv** toolbox to check your answers.

Listing 4.2: SMV model

```
MODULE main

VAR
  s : {s1, s2, s3, s4, s5};
  a : boolean;
  b : boolean;
  c : boolean;
ASSIGN
  init(s) := {s1, s2};

  next(s) := case
    s = s1 : {s3, s4};
    s = s2 : s4;
    s = s3 : s4;
    s = s4 : {s2, s3, s5};
    s = s5 : {s4, s5};
  esac;

  a := case
    s = s1 : TRUE;
    s = s5 : TRUE;
    TRUE   : FALSE;
  esac;

  b := case
    s = s3 : TRUE;
    s = s4 : TRUE;
    s = s5 : TRUE;
    TRUE   : FALSE;
  esac;

  c := case
    s = s2 : TRUE;
    s = s3 : TRUE;
    s = s5 : TRUE;
    TRUE   : FALSE;
  esac;
```

**Exercise 4.4.** Let the following transition system $TS$ be given.




For each of the following formulas find all states that satisfy the given formula.

a) $\mathsf{X}\mathsf{G}\neg a$

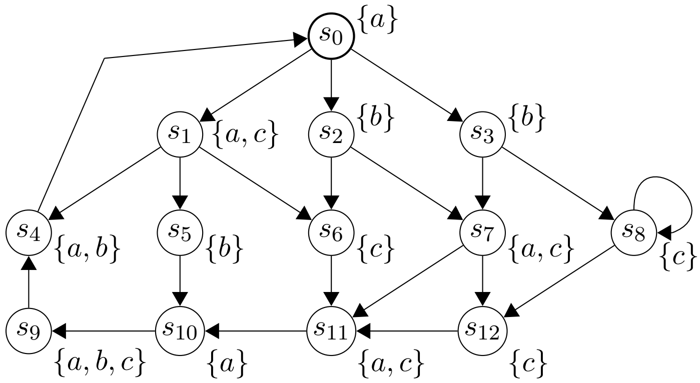
b) $b\,\mathsf{W}\,c$

c) $\mathsf{FX}a \vee \mathsf{FG}c$

d) $c\,\mathsf{R}\,b$

Develop an **SMV** model for the transition system and check your answers.

**Exercise 4.5.** Let the following transition system $TS$ be given ($I = \{s_0\}$).




Check whether $TS$ satisfies the following LTL formulas:

a) $\mathsf{GF}((b \vee c) \wedge \neg a)$

b) $\mathsf{X}(\neg a\,\mathsf{U}\,(a \wedge c)) \vee \mathsf{G}(\neg a \vee \neg c)$

c) $\mathsf{FX}c$

d) $\mathsf{X}(\neg a\,\mathsf{U}\,(a \wedge c))$

**Exercise 4.6.** Which of the following equivalences are correct? Prove the equivalence or provide a counterexample that illustrates that the formula on the left and the formula on the right are not equivalent ([1]).

a) $\mathsf{GG}(a \vee \neg b) \;\equiv\; \neg\mathsf{F}(\neg a \wedge b)$

b) $\mathsf{F}(a \wedge b) \;\equiv\; \mathsf{F}a \wedge \mathsf{F}b$

c) $\mathsf{F}a \wedge \mathsf{XG}b \;\equiv\; \mathsf{F}b$

d) $\mathsf{GF}a \Rightarrow \mathsf{GF}b \;\equiv\; \mathsf{G}(a \Rightarrow \mathsf{F}b)$

e) $\neg(a\,\mathsf{U}\,b) \;\equiv\; \neg b\,\mathsf{W}\,(\neg a \wedge \neg b)$

f) $\mathsf{XF}a \;\equiv\; \mathsf{FX}a$

g) $(\mathsf{FG}a) \wedge (\mathsf{FG}b) \;\equiv\; \mathsf{F}(\mathsf{G}a \wedge \mathsf{G}b)$

h) $(a\,\mathsf{U}\,b)\,\mathsf{U}\,b \;\equiv\; a\,\mathsf{U}\,b$

# Chapter 5

# CTL logic

**Exercise 5.1.** Which of the following formulas are legal CTL formulas? Please justify the answer.

a) $\mathsf{AG}a \vee \mathsf{FE}b$

b) $(a \vee \mathsf{X}a) \Rightarrow \mathsf{EG}(a \vee b)$;

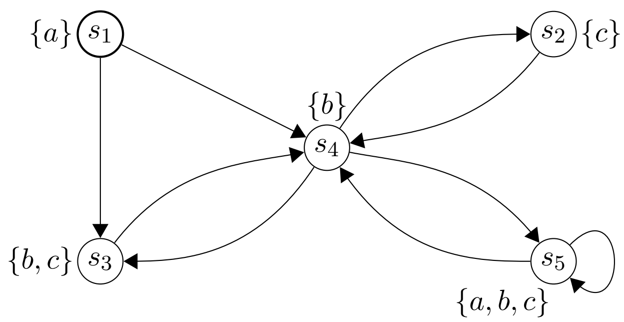
c) $\mathsf{AXAX}a$;

d) $\mathsf{AEX}a$;

e) $a \vee b$;

f) $a \vee b \,\mathsf{U}\, c$.

**Exercise 5.2.** Let the following transition system $TS$ be given ($I = \{s_1\}$):




Check (do not use **nuXmv**) whether $TS$ satisfies the following CTL formulas:

a) $\mathsf{AXEG}\neg a$

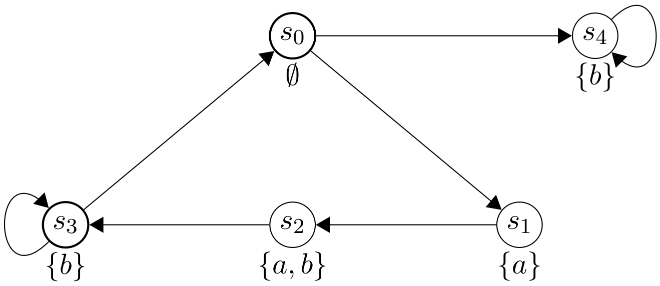
b) $\mathsf{A}(\neg(a \wedge b) \,\mathsf{U}\, (\neg a \wedge c)$

c) $\mathsf{EXEG}\neg a$

d) $\mathsf{AXE}((b \vee c) \,\mathsf{U}\, (a \wedge b \wedge c))$.

Use **nuXmv** to check your answers.

**Exercise 5.3.** Let the following transition system $TS$ be given ([1]):




Determine the satisfaction sets $Sat(\Phi_i)$ for the following formulas:

$\Phi_1 = \mathsf{A}(a\,\mathsf{U}\,b) \vee \mathsf{EX}(\mathsf{AG}\,b)$

$\Phi_2 = \mathsf{AGA}(a\,\mathsf{U}\,b)$

$\Phi_3 = (a \wedge b) \Rightarrow \mathsf{EGEXA}(b\,\mathsf{W}\,a)$

$\Phi_4 = (\mathsf{AGEF}a)$

**Exercise 5.4.** Which of the following formulas are legal LTL or CTL formulas? Please justify the answer.

a) $\mathsf{AG}a \vee \mathsf{E}(b\,\mathsf{U}\,a)$
b) $(a \vee \mathsf{AX}a) \Rightarrow \mathsf{EF}(a \vee b)$;
c) $\mathsf{FXXX}a$;
d) $a$;
e) $\mathsf{G}a \vee \mathsf{F}b$;
f) $a \vee b\,\mathsf{U}\,c$.

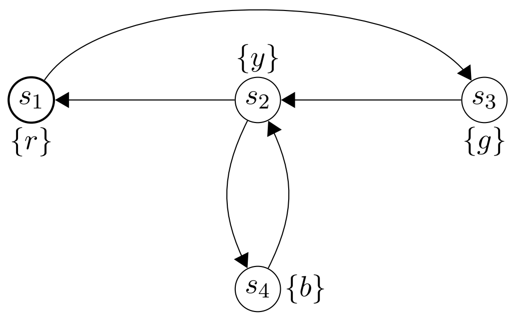
**Exercise 5.5.** Provide an example of a transition system (up to 5 states) that satisfies the CTL formula, but does not satisfy the LTL formula.

a) $\mathsf{AX}\;\mathsf{E}(a\,\mathsf{U}\,(b \vee c))$, $\quad \mathsf{X}\;(a\,\mathsf{U}\,(b \vee c))$
b) $(\mathsf{EG}\;\mathsf{AX}a) \wedge b$, $\quad (\mathsf{G}\;\mathsf{X}a) \wedge b$

**Exercise 5.6.** Let a transition system with $AP = \{a, b, c\}$ be given. Define the following properties using CTL logic.

a) Each execution of the system leads to a state in which $a$ is satisfied and such that in any subsequent state $b$ is not satisfied.
b) It is possible an execution that leads to a state in which $a$ is satisfied and in the next three states only $b$ or $c$ is satisfied.
c) For any state, if $a$ is satisfied then $b$ is satisfied in the next state.
d) Each proposition belonging to $AP$ is satisfied infinitely often.
e) Each execution of the system leads to a state in which $a$ is satisfied and such that in all previous states $b$ is satisfied and $c$ is not satisfied.

**Exercise 5.7.** Let the following transition system $TS$ be given ([1]).




The system represents a traffic light that is able to blink yellow. Signals:

- $r$ – red,
- $y$ – yellow,
- $g$ – green,
- $b$ – blinking yellow.

For each of the following formulas find all states that satisfy the given formula.

a) $\mathsf{AF}y$
b) $\mathsf{AG}y$
c) $\mathsf{AGAF}y$
d) $\mathsf{AF}g$
e) $\mathsf{EF}g$
f) $\mathsf{EG}g$
g) $\mathsf{EG}\neg g$
h) $\mathsf{A}(b \,\mathsf{U}\, \neg b)$
i) $\mathsf{E}(b \,\mathsf{U}\, \neg b)$
j) $\mathsf{A}(\neg b \,\mathsf{U}\, \mathsf{EF}b)$
k) $\mathsf{A}(g \,\mathsf{U}\, \mathsf{A}(y \,\mathsf{U}\, r))$
l) $\mathsf{A}(\neg b \,\mathsf{U}\, b)$

**Exercise 5.8.** Provide an example of a transition system (up to 5 states) that satisfies one of the specified formulas but does not satisfy the other.

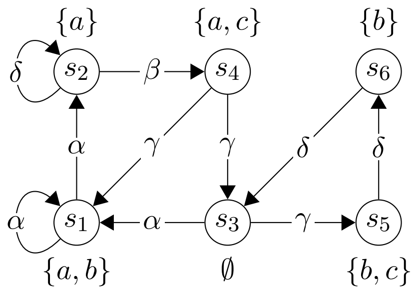
a) $\mathsf{AX}\ (\neg b \wedge \mathsf{EG}a)$, $\quad$ $\mathsf{X}\ (\neg b \wedge \mathsf{G}a)$

b) $b \wedge c \wedge (\mathsf{AG}\ \mathsf{EX}a)$, $\quad$ $b \wedge c \wedge (\mathsf{G}\ \mathsf{X}a)$

**Exercise 5.9.** Which of the following assertions are correct? Justify the answer ([1]).

a) If $s \models \mathsf{EG}a$, then $s \models \mathsf{AG}a$.
b) If $s \models \mathsf{AF}a \vee \mathsf{AF}b$, then $s \models \mathsf{AF}(a \vee b)$.
c) If $s \models \mathsf{AF}a \vee \mathsf{AF}b$, then $s \models \mathsf{EF}(a \wedge b)$.
d) If $s \models \mathsf{A}(a \,\mathsf{U}\, b)$, then $s \models \neg(\mathsf{E}(\neg b \,\mathsf{U}\, (\neg a \wedge \neg b)) \vee \mathsf{EG}\neg b)$.

**Exercise 5.10.** Let the following transition system $TS$ be given ($I = \{s_1\}$):




Check whether $TS$ satisfies the following CTL formulas:

a) $\mathsf{AX}(\neg b \wedge \mathsf{EG}a)$

b) $\mathsf{EG}a \vee \mathsf{E}(a\,\mathsf{U}\,\neg(a \wedge b \wedge c))$

c) $a \wedge (\mathsf{AX}a) \wedge (\mathsf{AX}\ \mathsf{AX}a) \wedge (\mathsf{AX}\ \mathsf{AX}\ \mathsf{AX}\ \mathsf{AX}a)$

d) $\mathsf{EF}\ \mathsf{EG}(\neg a)$

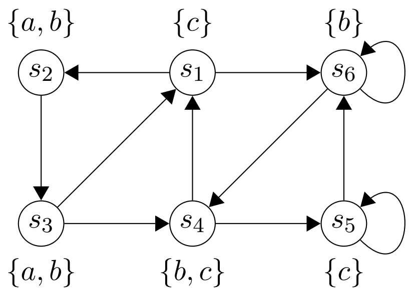
**Exercise 5.11.** Using the example from lectures (**see slide 49, part 2**) build a nuXmv model for solving the following logic puzzles.

A ferryman has to transport a sheep, two buffalo, and two tigers across a river. On each journey he can carry at most one item. However he cannot leave unattended on the same side of the river the sheep and a tiger (the tiger will eat the sheep) or one buffalo and two tigers (the tigers will eat the buffalo).

# Chapter 6

# RTCTL logic

**Exercise 6.1.** Let the following transition system $TS$ be given ($I = \{s_1\}$):




Check (do not use **nuXmv**) whether $TS$ satisfies the following RTCTL formulas:

a) $\mathsf{AX}\,\mathsf{AG}_{[1,2]}\neg a$

b) $\mathsf{AF}_{[0,3]}\mathsf{EG}c$

c) $\mathsf{A}((b \vee c)\,\mathsf{U}_{[0,4]}(b \wedge c))$

d) $\mathsf{E}((b \vee c)\,\mathsf{U}_{[3,3]}(b \wedge c))$

e) $\mathsf{EF}_{[2,3]}\mathsf{AX}a$

f) $\mathsf{EF}_{[5,5]}\mathsf{AX}a$

Use **nuXmv** to check your answers.

**Exercise 6.2.** Transform RTCTL formulas from the Exercise 6.1 into CTL formulas.

**Exercise 6.3.** Transform the following RTCTL formulas into CTL formulas.

a) $\mathsf{AX}\,\mathsf{AG}_{[1,\infty)}\neg a$

b) $\mathsf{AF}_{[0,\infty)}\mathsf{EG}c$

c) $\mathsf{A}(a\,\mathsf{U}_{[2,\infty)}b)$

d) $\mathsf{E}(a\,\mathsf{U}_{[3,\infty)}b)$

e) $\mathsf{EF}_{[2,\infty)}\mathsf{AX}a$

**Exercise 6.4.** Let a transition system with $AP = \{a, b, c\}$ be given. Define the following properties using RTCTL logic.

a) Each execution of the system leads within 20 time units to a state in which $a$ is satisfied and such that in any subsequent state $b$ is not satisfied for 10 time units.

b) It is possible an execution that leads to a state in which $a$ is satisfied and in the next three states only $b$ or $c$ is satisfied. Don't use X operator.

c) For any state, if $a$ is satisfied then $b$ is satisfied within 5 time units.

d) Each proposition belonging to $AP$ is satisfied at least once within the first 10 time units.

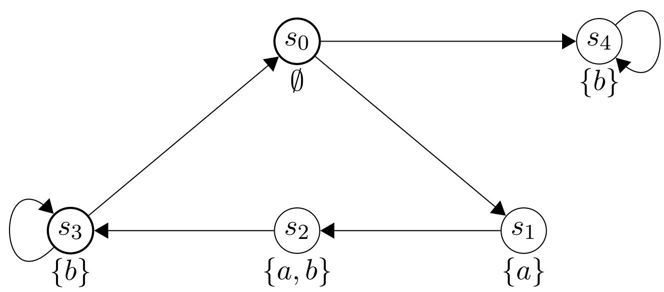
e) Each execution of the system leads within 30 time units to a state in which $a$ is satisfied and such that in all previous states $b$ is satisfied and $c$ is not satisfied.

**Exercise 6.5.** Provide an example of a transition system that satisfies one of the following formulas, but does not satisfy the other.

a) $\mathsf{AX}\,\mathsf{E}(a\,\mathsf{U}\,(b \vee c))$, $\quad \mathsf{AX}\,\mathsf{E}(a\,\mathsf{U}_{[0,3]}(b \vee c))$

b) $(\mathsf{EG}\,\mathsf{AX}a) \wedge b$, $\quad (\mathsf{EG}_{[0,3]}\mathsf{AX}a) \wedge b$

**Exercise 6.6.** Let the following transition system $TS$ be given ([1]):




Determine the satisfaction sets $Sat(\Phi_i)$ for the following formulas:

$\Phi_1 = \mathsf{A}(a\,\mathsf{U}_{[0,1]}b) \vee \mathsf{EX}(\mathsf{AG}_{[0,2]}b)$

$\Phi_2 = \mathsf{AG}_{[0,1]}\mathsf{A}(a\,\mathsf{U}_{[0,1]}b)$

$\Phi_3 = (\mathsf{AGEF}_{[0,2]}a)$

**Exercise 6.7.** Which of the following formulas are legal RTCTL formulas? Please justify the answer.

a) $\mathsf{AG}a \vee \mathsf{E}(b\,\mathsf{U}\,a)$

b) $(a \vee \mathsf{AX}_{[0,5]}a) \Rightarrow \mathsf{EF}(a \vee b)$;

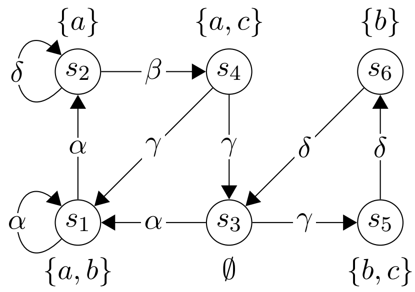
c) $\mathsf{EF}_{[3,3]}\mathsf{AX}\,\mathsf{AX}\,\mathsf{EX}a$;

d) $a$;

e) $\mathsf{AG}a \vee \mathsf{F}b$;

f) $a \vee \mathsf{E}(b\,\mathsf{U}_{[2,\infty)}c)$.

**Exercise 6.8.** Let the following transition system $TS$ be given ($I = \{s_1\}$):




Check whether $TS$ satisfies the following CTL formulas:

a) $\mathsf{AX}(\neg b \wedge \mathsf{EG}_{[0,2]} a)$

b) $\mathsf{EG} a \vee \mathsf{E}(a \,\mathsf{U}_{[1,2]} \neg(a \wedge b \wedge c))$

c) $\mathsf{AG}_{[0,3]} a$

d) $\mathsf{EF}_{[0,2]}\, \mathsf{EG}_{[1,\infty)}(\neg a)$

**Exercise 6.9.** Use the example from lectures (**see slide 49, part 2**) and:

a) Calculate the minimum and maximum time necessary to transport all items to the other side of the river.

b) Calculate the minimum and maximum time when the wolf and goat are not on the same side of the river.

**Exercise 6.10.** Which of the following assertions are correct? Justify the answer.

a) If $s \models \mathsf{E}(a \,\mathsf{U}_{[0,n]} b)$, then $s \models \mathsf{EF}_{[0,n]} a$.

b) If $s \models \mathsf{AF}_{[0,m]} a \vee \mathsf{AF}_{[0,n]} b$, then $s \models \mathsf{AF}_{[0,p]}(a \vee b)$, where $p = \min(m, n)$.

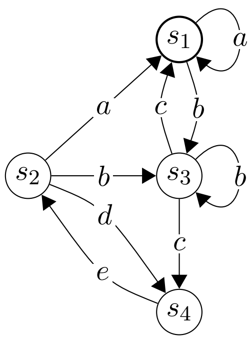
c) If $s \models \mathsf{AF}_{[0,m]} a \vee \mathsf{AF}_{[0,n]} b$, then $s \models \mathsf{AF}_{[0,p]}(a \wedge b)$, where $p = \max(m, n)$.

d) If $s \models \mathsf{A}(a \,\mathsf{U}_{[m,m]} b)$, then $s \models \mathsf{AG}_{[m,m]} b \wedge \mathsf{AG}_{[0,m-1]} a$, where $n > 0$.

# Chapter 7

# $\mu$ calculus

**Exercise 7.1.** Let the following transition system $TS$ be given ($I = \{s_1\}$, [3]):




Check whether $TS$ satisfies the following $\mu$ formulas:

a) $\langle a \rangle\, true$

b) $\langle c \rangle\, true$

c) $[c]\, false$

d) $[a]\, \langle b \rangle\, true$

e) $[a]\, \langle d \rangle\, true$

f) $[a]\, (\langle d \rangle\, true \vee \langle a \rangle\, \langle d \rangle\, true)$

g) $[a]\, [b]\, \langle c \rangle\, true$

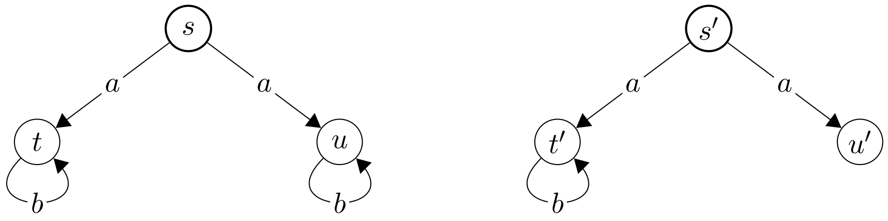
h) $[a]\, [b]\, ([b]\, false \Rightarrow \langle d \rangle\, true)$

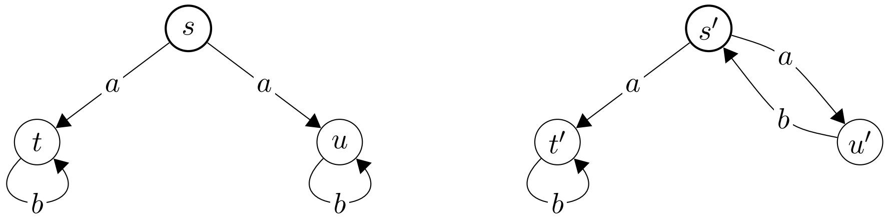
i) $[b]\, [c]\, ([b]\, false \Rightarrow \langle e \rangle\, true)$

**Exercise 7.2.** Build a transition system that satisfies the given formulas. Can you build a system consisting of only 3 states? ([3])

- $\langle a \rangle\, (\langle b \rangle\, true \wedge \langle c \rangle\, true)$
- $[a]\, [b]\, (\langle d \rangle\, true \wedge [d]\, \langle e \rangle\, true)$
- $\langle a \rangle\, \langle c \rangle\, \langle d \rangle\, true$

**Exercise 7.3.** For each pair of given transition systems, give a $\mu$ formula that is *true* for one system, and not for the other (two formulas for each pair, [3]).

a) 

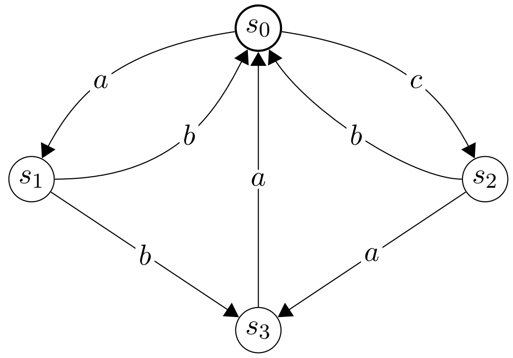

b) 


**Exercise 7.4.** Build a transition system that satisfies the formula $[a]\,\langle true.b\rangle\; true$, but does not satisfy the formula $[true^{*}.a]\,\langle true.b\rangle\; true$ ([3]).

**Exercise 7.5.** Let a set of actions $A = \{a, b, \dots\}$ be given. Define the following properties using $\mu$ logic.

a) $a$ never occurs.
b) $a$ should occur exactly once.
c) $a$ alternate with $b$ but there may be other actions between them.
d) $a$ alternate with $b$ (acceptable sequences of actions are: $abab\dots$, $baba\dots$);
e) After an occurrence of $a$, $b$ or next $a$ are only possible.
f) $a$ never occurs after $b$.
g) A single $b$ never occurs after the $a$ action (two or more consecutive $b$ actions are allowed).
h) The system can not start with action $b$.
i) The system can not start with two $b$ actions.
j) The deadlock is possible after two initial actions.
k) The deadlock is possible after the second occurrence of the $a$ action (the number of other actions is unlimited).
l) It is possible to execute a sequence of at least five actions such that it does not contain the $a$ action.

**Exercise 7.6.** Let the following transition system $TS$ be given.




Define $\mu$ formulas describing the following properties:

a) The system is deadlock free.

b) There is a path along which $a$ never occurs.

c) There is a path along which $a$ occurs exactly once.

d) Between two consecutive $c$ there is a $b$.

e) Two consecutive $a$ are not possible.

f) After an occurrence of $a$ or $c$, $b$ must occur in the next step.

g) After an occurrence of $a$ or $c$, occurrence of $b$ is inevitable in a finite number of steps.

h) The subsequence of actions $a, b, c$ never occurs.

Use **CADP evaluator** to check whether the transition system satisfies the properties.

Listing 7.1: Aldebaran format for the model

```
des (0,7,4)
(0,"a",1)
(0,"c",2)
(1,"b",0)
(1,"b",3)
(2,"b",0)
(2,"a",3)
(3,"a",0)
```

**Exercise 7.7.** Build a transition system with 5 states that satisfies the given formulas. Justify that all formulas are satisfied.

- $[\mathit{true}^*]\,\langle \mathit{true}\rangle\, \mathit{true}$
- $\langle \mathit{true}.b\rangle\, \mathit{true}$
- $[b]\,\mathit{false}$
- $\langle \mathit{true}^*.d\rangle\, \mathit{true}$
- $([\mathit{true}.d]\,\mathit{false}) \wedge ([d.\mathit{true}]\,\mathit{false})$
- $\nu X.(\langle a.b.c\rangle\, X)$
- $[\mathit{true}^*.d.(\neg a)]\,\mathit{false}$
- $[\mathit{true}^*.d]\,\nu X.(\langle a\rangle\, X)$

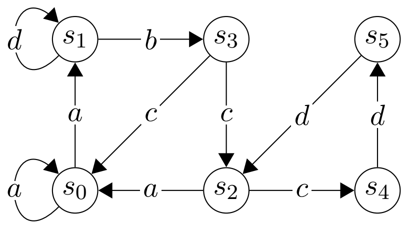
**Exercise 7.8.** Build a transition system with 4 states that satisfies the given formulas. Justify that all formulas are satisfied.

- $[\mathit{true}^*]\,\langle \mathit{true}\rangle\, \mathit{true}$
- $\langle c\rangle\, \mathit{true}$
- $\langle \mathit{true}^*.a\rangle\, \mathit{true}$
- $\langle \mathit{true}^*.d\rangle\, \mathit{true}$
- $[\mathit{true}^*.c.c]\,\mathit{false}$
- $\nu X.(\langle b\rangle\, X)$
- $[\mathit{true}^*.a]\,\nu X.(\langle b\rangle\, X)$
- $[\mathit{true}^*.d]\,\mu X.(\langle \mathit{true}\rangle\, \mathit{true} \wedge [\neg c]X)$

**Exercise 7.9.** Build a transition system with 4 states that satisfies the given formulas. Justify that all formulas are satisfied.

- $\nu X.(\langle b.c.b\rangle\ X)$
- $\nu X.(\langle a.c.b\rangle\ X)$
- $[true]\ \nu X.(\langle a\rangle\ X)$
- $[true^*.c.c]\ \mu X.(\langle true\rangle\ true \wedge [\neg a]X)$
- $[true^*.a.b]\ false$
- $\langle true^*.a.c\rangle\ true \wedge \langle true^*.c.a\rangle\ true$

**Exercise 7.10.** Let the following transition system $TS$ be given.




Use **CADP evaluator** to check whether the transition system satisfies the given properties.

a) `<true*."c"."c"> true`

b) `[true*."b".true."b"] false`

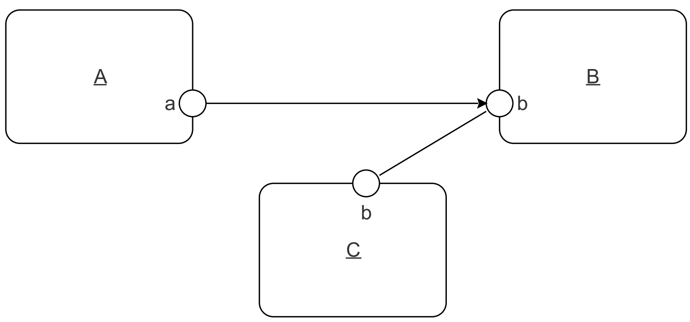
c) `[(not "b")."c".true*."b"] false`

d) `[true*."b"]<true*."a"> true`

e) `nu X.(<"a"."b"."c"> X)`

f) `[true*."b"] mu X.(<true>true and [not "a"]X)`

**Exercise 7.11.** Let the following Alvis model be given.




```
agent A {
  loop {
    out a;
  }
}
```

```
agent B {
  loop {
    in b;
    out b;
  }
}
```

```
agent C {
  loop {
    out b;
    in b;
  }
}
```

The transition system for the given model exported into Aldebaran format is stored in **t105.aut** file.

Define $\mu$ formulas describing the following properties and use **CADP evaluator** to check whether the transition system satisfies the properties.

a) The system is deadlock free.

b) An infinite behaviour (sequence of actions) is possible.

c) Each signal provided by agent $A$ ($out(A.a)$) is collected (if this happens occurrence of $loop(A)$ is inevitable in a finite number of steps).

d) There is an infinite path along which none action of agent $A$ occurs.

e) There is an infinite path along which none action of agent $B$ occurs.

f) There is an infinite path along which none action of agent $C$ occurs.

g) There is a path consisting of an infinite concatenation of subsequence: $loop(B)$, $loop(C)$, $in(B.b)$, $out(C.b)$, $out(B.b)$, $in(C.b)$.

# Chapter 8

# Sample exam

**1. (8 points)**
Define the following properties using $\mu$ logic (for items e)–h) use fixed point operators).

a) The system can start with at least three $b$ actions.

b) $c$ can not occur as the second action.

c) $c$ can not occur before occurring $b$ at least twice (not necessarily consecutive).

d) If the system starts with $b$, then it is deadlock free.

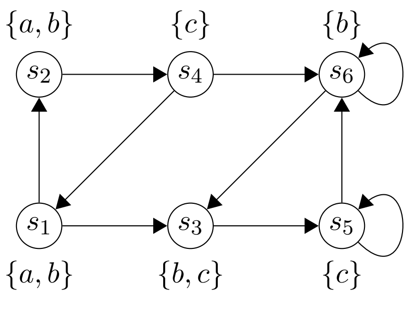
e) It is possible to execute an infinite sequence of actions such that it does not contain the $c$ action.

f) For any sequence of actions starting with $a.a.a$, occurrence of $b$ is inevitable in a finite number of steps.

g) For any sequence of actions starting with $a.a.a$, it is then possible to perform an infinite number of $b$ actions.

h) If the system starts with $b$, then it can only perform a finite number of consecutive $b$ actions.

**2. (8 points)**
Let the following transition system $TS$ be given ($I = \{s_1, s_6\}$).




Check whether $TS$ satisfies the following LTL formulas. Justify your answers.

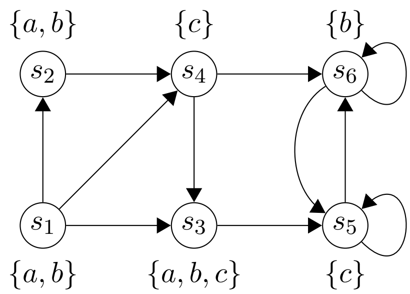
a) $\neg(a \wedge c) \wedge (\mathsf{X}\mathsf{G}\neg a)$

b) $b \,\mathsf{W}\, c$

c) $\mathsf{F}\mathsf{G}b \vee \mathsf{F}\mathsf{G}c$

d) $b \wedge (c \,\mathsf{R}\, b)$

**3. (8 points)**
Let the following transition system $TS$ be given ($I = \{s_1\}$).




Check whether $TS$ satisfies the following CTL formulas. Justify your answers.

a) $\mathsf{AXAXAG}(\neg a)$

b) $\mathsf{AGE}[(a \vee c)\,\mathsf{U}\,b]$

c) $(\neg c) \wedge (\mathsf{AGAF}c)$

d) $\mathsf{EGEF}a$

**4. (6 points)**
Consider the set of atomic propositions defined by $AP = \{a, b, c\}$. Characterize each of the following linear-time properties as being either an invariant, safety property, liveness property, or none of these. Justify your answers.

a) At least one property belonging to $AP$ is satisfied in each state.

b) $a$ never occurs in two consecutive states.

c) $a$, $b$, and $c$ never occur together.

d) $a$ and $c$ occur together at most twice.

e) $a$ occur infinitely many times.

f) If $a$ and $c$ occur in the initial state then $b$ is inevitable in future.

# Bibliography


[1] C. Baier and J.-P. Katoen. *Principles of Model Checking*. The MIT Press, London, UK, 2008.

[2] C. Fencott. *Formal Methods for Concurrency*. International Thomson Computer Press, Boston, MA, USA, 1995.

[3] J.J.A. Keiren. Modal $\mu$-calculus (version 1.1). Technical report, VU University Amsterdam, 2013.