

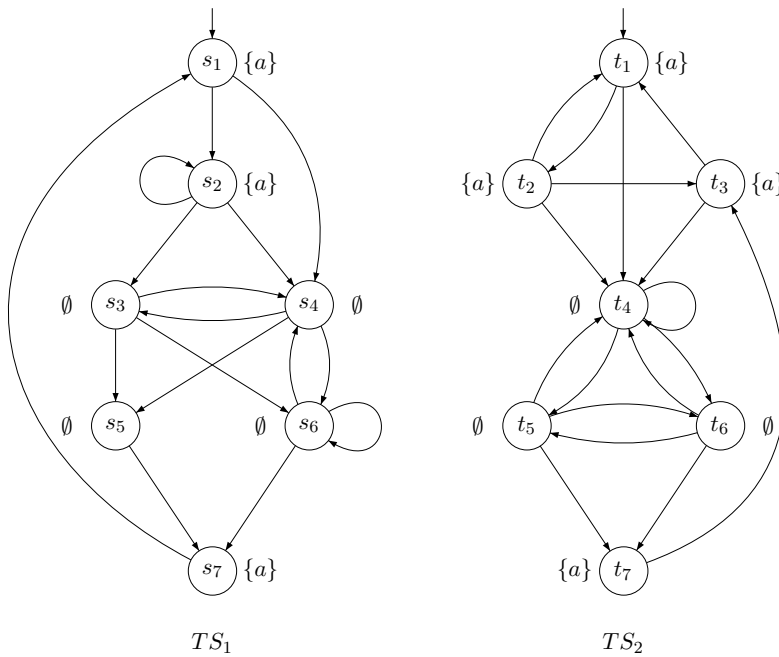
## Advanced Model Checking Summer term 2014

### – Solution 2 –

#### Exercise 1

(1 + 1 points)

- (a) Given transition systems  $TS_1$  and  $TS_2$ , determine whether  $Traces(TS_1) = Traces(TS_2)$ .

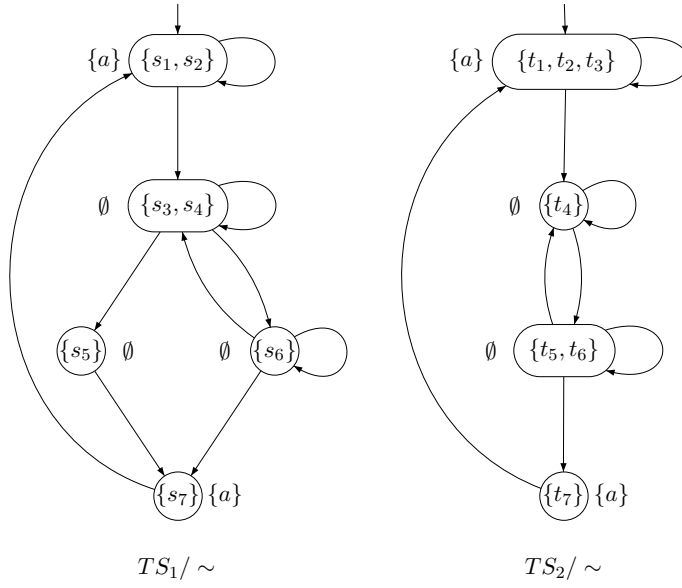


- (b) Provide two transition systems  $TS'_1$  and  $TS'_2$  such that  $Traces_{fin}(TS'_1) = Traces_{fin}(TS'_2)$ , but  $Traces(TS'_1) \neq Traces(TS'_2)$ .

#### Solution:

- (a) Although  $TS_1 \not\sim TS_2$ ,  $TS_1 \simeq TS_2$ . Additionally,  $TS_1$  and  $TS_2$  do not have terminal states, which implies  $Traces(TS_1) = Traces(TS_2)$ .

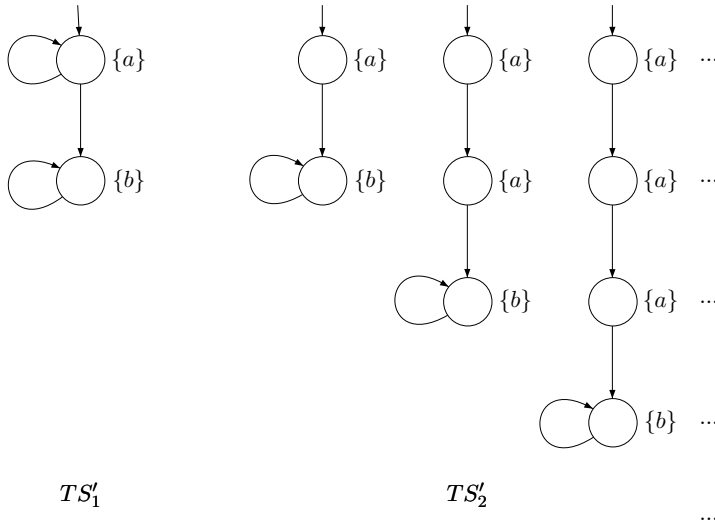
To show  $TS_1 \not\sim TS_2$  but  $TS_1 \simeq TS_2$ , we first compute the bisimulation quotient of  $TS_1$  and  $TS_2$ , i.e.,  $TS_1 / \sim$  and  $TS_2 / \sim$  as follows:



State  $\{s_5\}$  does not have a corresponding bisimilar state in  $TS_2/\sim$ , thus  $(TS_1/\sim) \not\sim (TS_2/\sim)$ . Since  $TS_1 \sim (TS_1/\sim)$  and  $TS_2 \sim (TS_2/\sim)$ , it holds that  $TS_1 \not\sim TS_2$ .

It is not hard to show that  $(TS_1/\sim) \simeq (TS_2/\sim)$ , due to the similar argument,  $TS_1 \simeq TS_2$ .

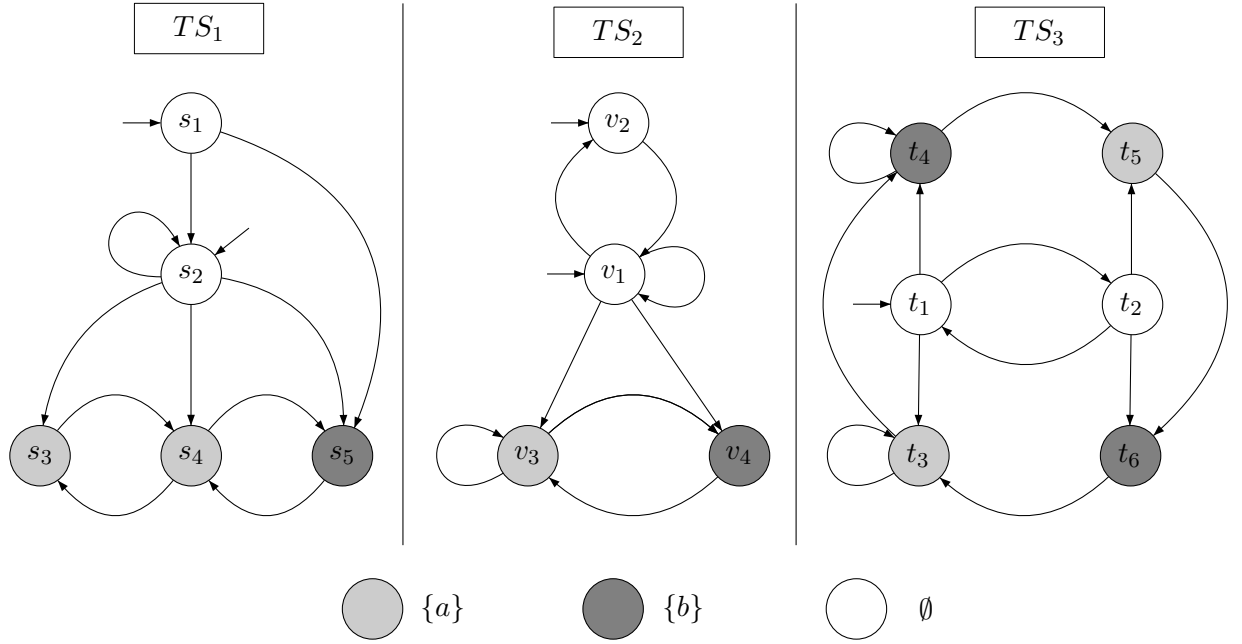
- (b)  $Traces_{fin}(TS'_1) = Traces_{fin}(TS'_2) = \{a^*, a^*b^*\}$ .  
 $Traces(TS'_1) = \{a^\omega, a^*b^\omega\}$ , but  $Traces(TS'_2) = \{a^*b^\omega\}$ .



## Exercise 2

(1 + 1 points)

Consider the transition systems  $TS_1$ ,  $TS_2$ ,  $TS_3$  over  $AP = \{a, b\}$  shown in the following figure:



### Questions:

- (a) for each  $i, j \in \{1 \dots 3\} \times \{1 \dots 3\}$ ,  $i \neq j$ , determine whether  $TS_i \preceq TS_j$
- (b) for each case  $TS_i \not\preceq TS_j$ , give a  $\forall\text{CTL}_{\setminus U}$  - formula that distinguishes  $TS_i$  and  $TS_j$ .

### Solution:

- (a) Try to construct the simulation relation  $R$  by the *Definition 7.46* of the Lecture notes:

(1)  $TS_1 \preceq TS_2$ : The simulation is  $R = \{(s_1, v_1), (s_2, v_1), (s_3, v_3), (s_4, v_3), (s_5, v_4)\}$ .

(2)  $TS_2 \not\preceq TS_3$ :

- State  $v_1$  should be simulated by  $t_1$  as an initial state.
- State  $v_1 \in \text{Post}(v_1)$  should be simulated by  $t_2 \in \text{Post}(t_1)$  as it is the only state from  $\text{Post}(t_1)$  such that  $L(v_1) = L(t_2)$ .
- Then similarly  $v_3 \in \text{Post}(v_1)$  should be simulated by  $t_5 \in \text{Post}(t_2)$ , which is not the case because  $v_3 \in \text{Post}(v_3)$  and there is no state in  $\text{Post}(t_5)$  labelled with  $L(v_3)$ .

(3)  $TS_1 \not\preceq TS_3$ :

- State  $s_2$  should be simulated by  $t_1$  as an initial state.
- State  $s_2 \in \text{Post}(s_2)$  should be simulated by  $t_2 \in \text{Post}(t_1)$  as it is the only state from  $\text{Post}(t_1)$  such that  $L(s_2) = L(t_2)$ .
- Then similarly  $s_3 \in \text{Post}(s_2)$  should be simulated by  $t_5 \in \text{Post}(t_2)$ , which is not the case because  $s_4 \in \text{Post}(s_3)$  and there is no state in  $\text{Post}(t_5)$  labelled with  $L(s_4)$ .

(4)  $TS_2 \not\preceq TS_1$ :

- State  $v_1$  should be simulated by  $s_1$  or  $s_2$  as an initial state.
- State  $s_1$  does not simulate  $v_1$  because for  $v_3 \in \text{Post}(v_1)$  and  $\text{Post}(s_1) = \{s_2, s_5\}$  we have  $L(v_3) \neq L(s_2)$  and  $L(v_3) \neq L(s_5)$ .
- For state  $s_2$  to simulate  $v_1$ , state  $v_3 \in \text{Post}(v_1)$  should be simulated by  $s_3$  or  $s_4 \in \text{Post}(s_2) = \{s_3, s_4, s_5\}$  (as  $L(v_3) = L(s_3) = L(s_4) \neq L(s_5)$ ).
- State  $v_3$  can not be simulated by  $s_3$  as there is no transition to the state labelled with  $\{b\}$ .
- State  $s_4$  can not simulate  $v_3$  because for  $v_3 \in \text{Post}(v_3)$  only  $s_3 \in \text{Post}(s_4)$  could simulate it due to the labelling, but we've seen in the previous point that it is not the case.

(5)  $TS_3 \not\preceq TS_1$ :

- State  $t_1$  should be simulated by either  $s_1$  or  $s_2$  as an initial state.
- State  $s_1$  does not simulate  $t_1$  because for  $t_3 \in Post(t_1)$  and  $Post(s_1) = \{s_2, s_5\}$  we have  $L(t_3) \neq L(s_2)$  and  $L(v_3) \neq L(s_5)$ .
- For state  $s_2$  to simulate  $t_1$ , state  $t_4 \in Post(t_1)$  should be simulated by  $s_5 \in Post(s_2) = \{s_3, s_4, s_5\}$  (as only  $L(t_4) = L(s_5)$ ).
- State  $t_4$  can not be simulated by  $s_5$  as  $t_4 \in Post(t_4)$  but there is no transition to the state labelled with  $\{b\}$  from  $s_5$ .

(6)  $TS_3 \not\preceq TS_2$ :

- State  $t_1$  should be simulated by either  $v_1$  or  $v_2$  as an initial state.
- State  $v_2$  does not simulate  $t_1$  because for  $t_3 \in Post(t_1)$  and  $Post(v_2) = \{v_1\}$  we have  $L(t_3) \neq L(v_1)$ .
- For state  $v_1$  to simulate  $t_1$ , state  $t_4 \in Post(t_1)$  should be simulated by  $v_4 \in Post(v_1) = \{v_3, v_4\}$  (as only  $L(t_4) = L(v_4)$ ).
- State  $t_4$  can not be simulated by  $v_4$  as  $t_4 \in Post(t_4)$  but there is no transition to the state labelled with  $\{b\}$  from  $v_4$ .

(b) Consider the following cases:

- (1)  $TS_1 \not\preceq TS_3$  **and**  $TS_2 \not\preceq TS_3$ : The following formula is valid in  $TS_3$  but not in  $TS_1$  and  $TS_2$ , it is targeted on distinguishing state  $t_5$  from states  $s_3, s_4, v_3$ .

$$\forall \bigcirc (\neg b \vee \forall \bigcirc (\neg a \vee \forall \bigcirc b))$$

- (2)  $TS_2 \not\preceq TS_1$ : The following formula is valid in  $TS_1$  but not in  $TS_2$ , it is targeted on distinguishing state  $s_3$  from state  $v_3$ .

$$\forall \bigcirc (\neg b \vee \forall \bigcirc \forall \bigcirc (\neg a \vee \forall \bigcirc a))$$

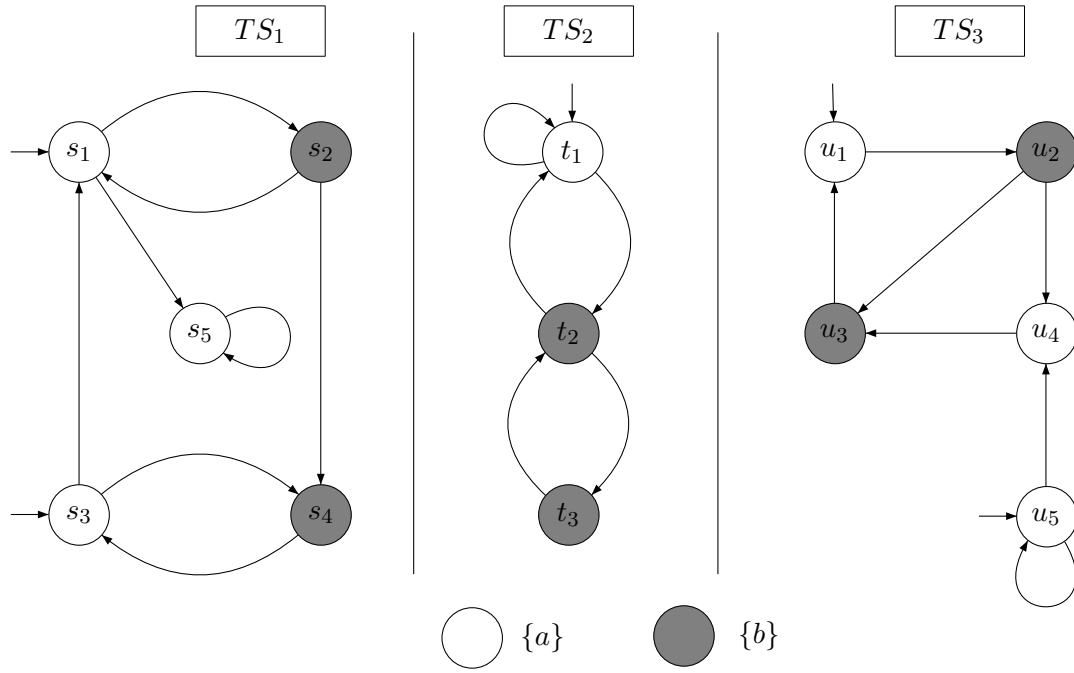
- (3)  $TS_3 \not\preceq TS_1$  **and**  $TS_3 \not\preceq TS_2$ : The following formula is valid in  $TS_1$  and  $TS_2$  but not in  $TS_3$ , it is targeted on distinguishing states  $s_5, v_4$  from state  $t_4$ .

$$\forall \bigcirc (\neg b \vee \forall \bigcirc a)$$

### Exercise 3

(2 points)

Consider three transitions systems given on the next Figure:



For each  $i, j \in \{1 \dots 3\} \times \{1 \dots 3\}$ ,  $i \neq j$ , determine whether  $TS_i \sqsubseteq TS_j$  or  $TS_i \not\sqsubseteq TS_j$ . Justify your answer.

**Solution:**

- $TS_2 \not\sqsubseteq TS_1$ , since  $ab^\omega \notin \text{Traces}(TS_2)$ , but there is no path in  $TS_1$  whose trace is stutter equivalent to  $ab^\omega$  as there is no  $b$ -cycle in  $TS_1$ .

$TS_1 \sqsubseteq TS_2$ . This can be seen as follows, States  $s_1$  and  $s_3$ , as well as  $s_2$  and  $s_4$  in  $TS_1$  can be grouped, yielding  $TS'_1$ , say, and it is easy to see that  $TS_1$  and  $TS'_1$  are stutter equivalence. We have  $\text{Traces}(TS'_1) = \{(ab)^\omega, (ab)^*a^\omega\}$ . It suffices to show that for each of these traces, there is a stutter-equivalent path in  $TS_2$ . For  $(ab)^\omega$  this is  $(t_1t_2)^\omega$ , whereas for  $(ab)^*a^\omega$  this is the path  $(t_1t_2)^*t_1^\omega$ .

- $TS_1 \not\sqsubseteq TS_3$ , since  $(ab)^*a^\omega \in \text{Traces}(TS_1)$ , but there is no  $a$ -cycle reachable once a  $b$ -state is visited in  $TS_3$ .

$TS_3 \sqsubseteq TS_1$ . This can be seen as follows, Traces in  $TS_3$  are of the form -ignoring stuttering either  $(ab)^\omega$  or  $a^\omega$ . The trace of path  $s_1s_5^\omega$  in  $TS_1$  is  $a^\omega$  whereas the trace of path  $(s_1s_2)^\omega$  equals  $(ab)^\omega$ .

- $TS_2 \not\sqsubseteq TS_3$ , since  $(ab)^*a^\omega \in \text{Traces}(TS_2)$ , but there is no  $a$ -cycle reachable once a  $b$ -state is visited in  $TS_3$ .

$TS_3 \sqsubseteq TS_2$ . By similar reasoning as for  $TS_3 \sqsubseteq TS_1$ .

#### Exercise 4

(4 points)

Let  $\varphi$  be an LTL formula such that  $\text{Word}(\varphi)$  is stutter insensitive.

Show that  $\varphi$  is equivalent to some  $\text{LTL}_{\setminus \bigcirc}$  formula  $\psi$ .

**Proof:** We are to show that for each LTL formula  $\varphi$  whose  $\text{Word}(\varphi)$  is stutter insensitive, we can define a transformation  $\tau : \text{LTL} \rightarrow \text{LTL}_{\setminus \bigcirc}$ , such that  $\varphi \equiv \tau(\varphi)$ . Stated differently, we will show that for every LTL formula  $\varphi$  there exists a  $\text{LTL}_{\setminus \bigcirc}$  formula  $\tau(\varphi)$  that agrees with  $\varphi$  for all traces:

$$\sigma \models \varphi \quad \text{iff} \quad \sigma \models \tau(\varphi) \quad \text{for all traces } \sigma.$$

Let  $\psi := \tau(\varphi)$ , we then obtain the  $\text{LTL}_{\setminus \bigcirc}$  formula  $\psi$ . The proof goes by induction on the structure of  $\varphi$ .

*Induction base.* For  $p \in AP$  we can simply set  $\tau(p) = p$ .

*Induction steps.*

- Case  $\varphi = \neg\varphi'$ . Set  $\tau(\varphi) = \neg\tau(\varphi')$ .
- Case  $\varphi = \varphi' \wedge \varphi''$ . Set  $\tau(\varphi) = \tau(\varphi') \wedge \tau(\varphi'')$ .
- Case  $\varphi = \varphi' \mathbf{U} \varphi''$ . Set  $\tau(\varphi) = \tau(\varphi') \mathbf{U} \tau(\varphi'')$ .
- Case  $\varphi = \bigcirc\varphi'$ . The situation is more difficult.

*Definition 1:*

A trace  $\sigma$  to be *stutter-free* if either

- $\sigma[i] \neq \sigma[i+1]$  for  $i \geq 0$  or
- there exists a  $k \geq 0$  such that  $\sigma[i] \neq \sigma[i+1]$  for  $i < k$  and  $\sigma[k] = \sigma[k+1] = \sigma[k+2] = \dots$

Obviously, there is exactly one stutter-free trace in each class of stutter-equivalent traces. Also, every suffix of a stutter-free trace is stutter-free.

*Lemma 1:*

Let  $\psi$  be a  $\text{LTL}_{\bigcirc}$  formula, then  $\text{Word}(\psi)$  is stutter-insensitive.

*Lemma 2:*

Let  $\psi$  be a  $\text{LTL}_{\bigcirc}$  formula, then

$$\sigma \models \psi \quad \text{iff} \quad \sigma \in \text{Word}(\psi) \quad \text{for all stutter-free } \sigma$$

Due to Lemma 1 and Lemma 2, it suffices to show that for every LTL formula  $\varphi$  there exists a  $\text{LTL}_{\bigcirc}$  formula  $\tau(\varphi)$  that agrees with  $\varphi$  for all stutter-free traces:

$$\sigma \models \varphi \quad \text{iff} \quad \sigma \models \tau(\varphi) \quad \text{for all stutter-free } \sigma.$$

Assume  $AP = \{p_0, \dots, p_{n-1}\}$ . Let  $N : AP^n \rightarrow \{\text{true}, \text{false}\}^n$  be the set of all possible valuations of the atomic proposition set. For each  $\nu \in N$ , let  $\beta_\nu$  be the formula  $\alpha_0 \wedge \dots \wedge \alpha_{n-1}$  where

$$\alpha_j = \begin{cases} p_j & \text{if } \nu(p_j) = \text{true} \\ \neg p_j & \text{if } \nu(p_j) = \text{false} \end{cases}$$

We have the following two observations which are crucial:

*Observation 1:*

If  $\nu, \nu' \in N$  and  $\nu \neq \nu'$ , then

$$\sigma \models \beta_\nu \wedge \bigcirc\beta_{\nu'} \quad \text{iff} \quad \sigma \models \beta_\nu \mathbf{U} \beta_{\nu'} \quad \text{for all stutter-free } \sigma \text{ over } AP.$$

*Observation 2:*

$\sigma \models \bigcirc\varphi'$  if and only if either

- $\sigma[0] = \sigma[1] = \dots$  and  $\sigma[0] \models \varphi'$  or
- $\sigma[0] \neq \sigma[1]$  and  $\sigma \downarrow_1 \models \varphi'$ , where  $\sigma \downarrow_1 = \sigma[1]\sigma[2]\dots$  and  $\sigma \downarrow_1$  is stutter-free itself.

Thus, we can set

$$\tau(\bigcirc\varphi') = \bigvee_{\nu \in N} \left( \left( \bigwedge \beta_\nu \wedge \tau(\varphi') \right) \vee \bigvee_{\nu \neq \nu'} \left( \beta_\nu \mathbf{U} (\beta_{\nu'} \wedge \tau(\varphi')) \right) \right)$$

□