Prof. Dr. Ir. Dr. h. c. Joost-Pieter Katoen
Christian Hensel, Matthias Volk

Introduction to Model Checking (Summer Term 2018)

— Solution 10 (due 9th July) —

General Remarks

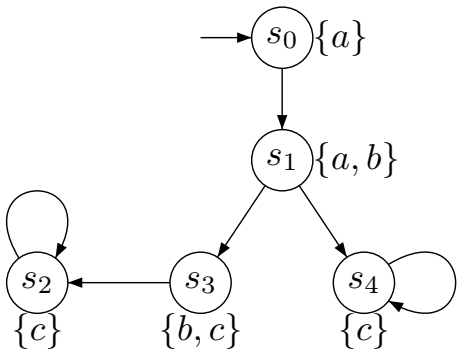
- The exercises are to be solved in groups of *three* students.
- You may hand in your solutions for the exercises just before the exercise class starts at 12:15 or by dropping them into the "Introduction to Model Checking" box at our chair *before 12:00*. Do *not* hand in your solutions via L2P or via e-mail.
- If a task asks you to justify your answer, an explanation of your reasoning is sufficient. If you are required to prove a statement, you need to give a *formal* proof.

Exercise 1 (6 Points)

Consider the following CTL-formulae

$$\Phi_1 = \exists\Diamond\forall\Box\, c \quad \text{and} \quad \Phi_2 = \forall(a \,\mathsf{U}\, \forall\Diamond\, c)$$

and the transition system outlined on the right. Decide whether $TS \models \Phi_i$ for $i = 1, 2$ using the CTL model checking algorithm from the lecture. Do not forget to translate to existential normal form and compute the satisfaction sets for subformulae.



Solution:

We apply the CTL model checking algorithm from the lecture.

- First consider the formula $\Phi_1 = \exists\Diamond\forall\Box\, c$:
  It can be expressed equivalently in ENF:

$$\begin{aligned}\Phi_1 &= \exists\Diamond\forall\Box\, c \\ &\equiv \exists\big(\mathit{true}\,\mathsf{U}\,\forall\Box\, c\big) \\ &\equiv \exists\big(\mathit{true}\,\mathsf{U}\,\neg\exists\Diamond\,\neg c\big) \\ &\equiv \exists\big(\mathit{true}\,\mathsf{U}\,\neg\exists(\mathit{true}\,\mathsf{U}\,\neg c)\big)\end{aligned}$$

  The bottom up computation of the satisfaction sets yields:
  - $Sat(\mathit{true}) = S$
  - $Sat(c) = \{s_2, s_3, s_4\}$
  - $Sat(\neg c) = \{s_0, s_1\}$
  - $Sat\big(\exists(\mathit{true}\,\mathsf{U}\,\neg c)\big)$ yields a backward search as follows:

* $E = T = Sat(\neg c) = \{s_0, s_1\}$
* Choose s_0: As $Pre(s_0) = \emptyset$, we get $E = \{s_1\}$, $T = \{s_0, s_1\}$.
* Choose s_1: $Pre(s_1) = \{s_0\}$. But $s_0 \notin Sat(true) \setminus T$ (i.e. it has already been visited), we get $E = \emptyset$, $T = \{s_0, s_1\}$.

$\implies T = \{s_0, s_1\}$.

– $Sat\big(\neg\exists(true \,\mathsf{U}\, \neg c)\big) = S \setminus Sat\big(\exists(true \,\mathsf{U}\, \neg c)\big) = S \setminus \{s_0, s_1\} = \{s_2, s_3, s_4\}$

– $Sat\Big(\exists\big(true \,\mathsf{U}\, \neg\exists(true \,\mathsf{U}\, \neg c)\big)\Big)$ again yields a backward search:

* $E = T = Sat\big(\neg\exists(true \,\mathsf{U}\, \neg c)\big) = \{s_2, s_3, s_4\}$
* Choose s_2: $Pre(s_2) = \{s_2, s_3\}$, but all predecessors are already in T. $\implies E = \{s_3, s_4\}$, $T = \{s_2, s_3, s_4\}$.
* Choose s_3: $Pre(s_3) = \{s_1\}$. Here we have $s_1 \in Sat(true)$ and $s_1 \notin T$. Therefore $T = T \cup \{s_1\} = \{s_1, s_2, s_3, s_4\}$ and $E = E \cup \{s_1\} = \{s_1, s_4\}$ (s_3 gets removed from E).
* Choose s_1: $Pre(s_1) = \{s_0\}$. Again $s_0 \in Sat(true) \setminus T$ and therefore $T = T \cup \{s_0\} = S$ and $E = E \cup \{s_0\} = \{s_0, s_4\}$.
* Choose s_0: $Pre(s_0) = \emptyset$ and we directly continue with s_4 ($E = \{s_0\}, T = S$).
* Choose s_4: $Pre(s_4) = \{s_1, s_4\}$ but s_1, s_4 are already in T. Therefore we stop with $E = \emptyset$.

$\implies T = \{s_0, s_1, s_2, s_3, s_4\} = S$

Therefore we have $Sat(\Phi_1) = \{s_0, s_1, s_2, s_3, s_4\}$ and $S_0 \subseteq Sat(\Phi_1) \implies TS \models \Phi_1$

- $\Phi_2 = \forall(a \,\mathsf{U}\, \forall\Diamond\, c)$. First, we transform Φ_2 into ENF:

$$\begin{aligned}
\forall(a \,\mathsf{U}\, \forall\Diamond\, c) &\equiv \neg\exists\big((\neg\forall\Diamond\, c) \,\mathsf{U}\, (\neg a \wedge \neg\forall\Diamond\, c)\big) \wedge \neg\exists\Box\, \neg\forall\Diamond\, c \\
&\equiv \neg\exists\big((\exists\Box\, \neg c) \,\mathsf{U}\, (\neg a \wedge \exists\Box\, \neg c)\big) \wedge \neg\exists\Box\, \exists\Box\, \neg c \\
&\equiv \neg\exists\big((\exists\Box\, \neg c) \,\mathsf{U}\, (\neg a \wedge \exists\Box\, \neg c)\big) \wedge \neg\exists\Box\, \neg c \qquad (*\exists\Box\, \exists\Box\, \Psi \equiv \exists\Box\, \Psi*)
\end{aligned}$$

– $Sat(a) = \{s_0, s_1\}$ and $Sat(c) = \{s_2, s_3, s_4\}$.

– $Sat(\neg a) = \{s_2, s_3, s_4\}$ and $Sat(\neg c) = \{s_0, s_1\}$.

– $Sat(\exists\Box\, \neg c)$ is computed by a backward search starting in the $S \setminus Sat(\neg c)$ states:

* $E = S \setminus Sat(\neg c) = \{s_2, s_3, s_4\}$ and $T = Sat(\neg c) = \{s_0, s_1\}$.
* The counter array is initialized according to TS as follows:

$$\begin{aligned}
&c[s_0] := 1 && (*\ s_0 \text{ has only one successor } *) \\
&c[s_1] := 2 && (*\ s_3 \text{ and } s_4 \text{ are successors of } s_1 \ *)
\end{aligned}$$

* Choose $s_2 \in E$ and check for each of its predecessors $Pre(s_2) = \{s_2, s_3\}$ whether it belongs to T.
  This is not the case, therefore we only remove s_2, $E = \{s_3, s_4\}, T = \{s_0, s_1\}$.
* Choose $s_3 \in E$: We have $s_1 \in Pre(s_3) \cap T$. Therefore with s_3, we have found one successor of s_1 which violates $\neg c$. We decrement the number of possible successors of the state s_1

$$c[s_1] := c[s_1] - 1 = 2 - 1 = 1$$

  and remove s_3 from the set E: $E = \{s_4\}$, $T = \{s_0, s_1\}$.
* Choose $s_4 \in E$: We have $Pre(s_4) = \{s_1, s_4\}$ and $s_1 \in T$. $\implies c[s_1] := c[s_1] - 1 = 0$ and $E = \emptyset$. As $c[s_1] = 0$, s_1 is removed from T (all successor states violate $\neg c$) and included in E: $E = \{s_1\}$, $T = \{s_0\}$.
* Choose $s_1 \in E$ with $Pre(s_1) = \{s_0\}$. As $s_0 \in T$, we decrement its counter $c[s_0] := 1 - 1 = 0$ and (because no successor state is left that could constitute a "good" path) remove it from $T = T \setminus \{s_0\} = \emptyset$. Then, s_0 is included in the set of "bad" states $E = \{s_0\}$.

$*$ Choose $s_0 \in E$ with $Pre(s_0) = \emptyset$. Nothing changes and $E = \emptyset$, $T = \emptyset$.

The result is $Sat(\exists\Box\,\neg c) = T = \emptyset$.

- $Sat(\neg\exists\Box\,\neg c) = S \setminus Sat(\exists\Box\,\neg c) = S \setminus \emptyset = S$.
- $Sat(\neg a \wedge \exists\Box\,\neg c) = Sat(\neg a) \cap Sat(\exists\Box\,\neg c) = \{s_2, s_3, s_4\} \cap \emptyset = \emptyset$.
- $Sat\big(\exists(\exists\Box\,\neg c)\ \mathsf{U}\ (\neg a \wedge \exists\Box\,\neg c)\big)$ invokes the $\exists\Psi_1\ \mathsf{U}\ \Psi_2$ backward search algorithm which directly terminates, because $E = Sat(\neg a \wedge \exists\Box\,\neg c) = \emptyset$.
  $\Longrightarrow T = \emptyset$
- $Sat\Big(\neg\exists\big((\exists\Box\,\neg c)\ \mathsf{U}\ (\neg a \wedge \exists\Box\,\neg c)\big)\Big) = S \setminus Sat\Big(\exists\big((\exists\Box\,\neg c)\ \mathsf{U}\ (\neg a \wedge \exists\Box\,\neg c)\big)\Big) = S \setminus \emptyset = S$
- $Sat(\Phi_2) = Sat\Big(\neg\exists\big((\exists\Box\,\neg c)\ \mathsf{U}\ (\neg a \wedge \exists\Box\,\neg c)\big)\Big) \cap Sat(\neg\exists\Box\,\neg c) = S \cap S = S$

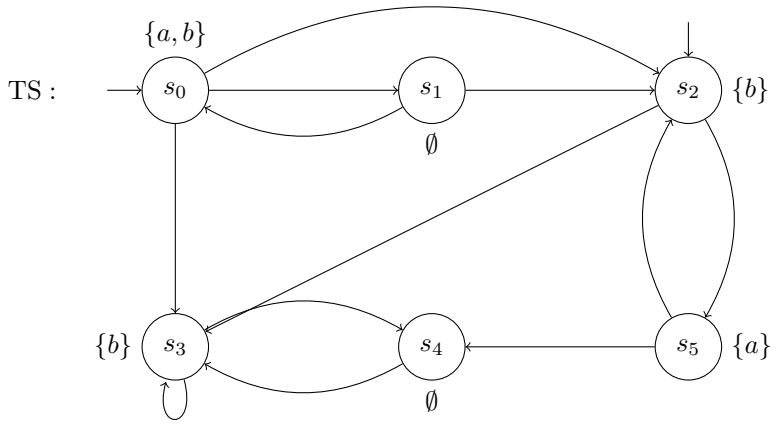
$\Longrightarrow S_0 \subseteq Sat(\Phi_2) \Longrightarrow TS \models \Phi_2$

Exercise 2* (3+2+3 Points)

Consider the following CTL formula Φ and the fairness assumption *sfair*:

$$\Phi = \forall\Box\forall\Diamond\, a$$
$$sfair = \Box\Diamond \underbrace{(b \wedge \neg a)}_{\Phi_1} \rightarrow \Box\Diamond \underbrace{\exists\Big(b\ \mathsf{U}\ (a \wedge \neg b)\Big)}_{\Psi_1}$$

and transition system TS over AP $= \{a, b\}$ which is given below.



Here, we abstract from the actions in TS as they are not relevant to the task.

(a) Determine $Sat(\Phi_1)$ and $Sat(\Psi_1)$ (without fairness). Justify your answer.

(b) Determine $Sat_{sfair}(\exists\Box\, true)$. Justify your answer.

(c) Determine $Sat_{sfair}(\Phi)$. Justify your answer.

Solution:

(a) We determine the satisfaction sets without fairness.

- Consider $\Phi_1 = b \wedge \neg a$.
  - $Sat(b) = \{s_0, s_2, s_3\}$
  - $Sat(a) = \{s_0, s_5\}$
  - $Sat(\neg a) = S \setminus Sat(a) = \{s_1, s_2, s_3, s_4\}$

  - $Sat(\Phi_1) = Sat(b) \cap Sat(\neg a) = \{s_0, s_2, s_3\} \cap \{s_1, s_2, s_3, s_4\} = \{s_2, s_3\}$
- Consider $\Psi_1 = \exists\big(b \,\mathsf{U}\, (a \wedge \neg b)\big)$.
  - $Sat(a) = \{s_0, s_5\}$
  - $Sat(b) = \{s_0, s_2, s_3\}$
  - $Sat(\neg b) = S \setminus Sat(b) = S \setminus \{s_0, s_2, s_3\} = \{s_1, s_4, s_5\}$
  - $Sat(a \wedge \neg b = Sat(a) \cap Sat(\neg b) = \{s_0, s_5\} \cap \{s_1, s_4, s_5\} = \{s_5\}$
  - $Sat(\Psi_1)$ yields a backwards search:
    * $E = T = Sat(a \wedge \neg b) = \{s_5\}$.
    * Choose $s_5 \in E$: $Pre(s_5) = \{s_2\}$. As $s_2 \in Sat(b) \setminus T = \{s_0, s_2, s_3\} \setminus \{s_5\} = \{s_0, s_2, s_3\}$ we add s_2: $E = \{s_2\}$, $T = \{s_2, s_5\}$.
    * Choose $s_2 \in E$: $Pre(s_2) = \{s_0, s_1, s_5\}$. As $s_0 \in Sat(b) \setminus T = \{s_0, s_2, s_3\} \setminus \{s_2, s_5\} = \{s_0, s_3\}$ we add s_0: $E = \{s_0\}$, $T = \{s_0, s_2, s_5\}$.
    * Choose $s_0 \in E$: $Pre(s_0) = \{s_1\}$. As $s_1 \notin Sat(b) \setminus T = \{s_0, s_2, s_3\} \setminus \{s_0, s_2, s_5\} = \{s_3\}$ we do not add s_0: $E = \emptyset$, $T = \{s_0, s_2, s_5\}$.

    Therefore $Sat(\Psi_1) = \{s_0, s_2, s_5\}$.

(b) We determine $Sat_{sfair}(\exists\Box\, true)$ and have to consider three SCCs: $\{s_0, s_1\}$, $\{s_3, s_4\}$ and $\{s_2, s_5\}$.

- Initially $T = \emptyset$.
- Consider the first SCC $\{s_0, s_1\}$. As $s_0 \in Sat(\Psi_1)$ the function $CheckFair$ returns $true$.
  $\Longrightarrow T = T \cup \{s_0, s_1\} = \{s_0, s_1\}$
- Consider the second SCC $\{s_3, s_4\}$. It is $s_3, s_4 \notin Sat(\Psi_1)$. Next we remove all states satisfying Φ_1 ($Sat(\Phi_1) = \{s_2, s_3\}$) in the SCC. The resulting graph is acyclic as only the state s_4 remains. Therefore the function $CheckFair$ returns $false$.
  $\Longrightarrow T = \{s_0, s_1\}$
- Consider the third SCC $\{s_2, s_5\}$. As $s_2, s_5 \in Sat(\Psi_1)$ the function $CheckFair$ returns $true$.
  $\Longrightarrow T = T \cup \{s_2, s_5\} = \{s_0, s_1, s_2, s_5\}$

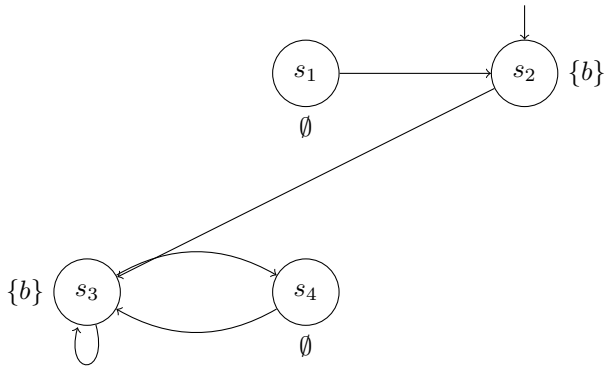
Therefore all states except s_3, s_4 satisfy the fairness constraints: $Sat_{sfair}(\exists\Box\, true) = \{s_0, s_1, s_2, s_5\}$.

(c) First, we convert Φ into ENF:

$$\begin{aligned}\Phi &= \forall\Box\forall\Diamond\, a \\ &\equiv \neg\exists\Diamond\exists\Box\neg a \\ &\equiv \neg\exists(true \,\mathsf{U}\, \exists\Box\neg a)\end{aligned}$$

We determine the satisfaction sets with respect to fairness.

- $Sat_{sfair}(true) = Sat(true) = \{s_0, s_1, s_2, s_3, s_4, s_5\}$
- $Sat_{sfair}(a) = Sat(a) = \{s_0, s_5\}$
- $Sat_{sfair}(\neg a) = S \setminus Sat_{sfair}(a) = S \setminus \{s_0, s_5\} = \{s_1, s_2, s_3, s_4\}$
- For $Sat_{sfair}(\exists\Box\neg a)$ we have to create the graph $G_{\neg a}$ which looks as follows:



The only SCC here is $\{s_3, s_4\}$ which does not satisfy the fairness constraint as shown in part (b).
Therefore $Sat_{sfair}(\exists\Box\,\neg a) = \emptyset$.

- $Sat_{sfair}\Big(\exists\big(true\ \mathsf{U}\ (\exists\Box\,\neg a)\big)\Big) = Sat\Big(\exists\big(true\ \mathsf{U}\ (\exists\Box\,\neg a \wedge a_{fair})\big)\Big) = \emptyset$
- $Sat_{sfair}(\Phi) = Sat_{sfair}\Big(\neg\exists\big(true\ \mathsf{U}\ (\exists\Box\,\neg a)\big)\Big) = S \setminus Sat_{sfair}\Big(\exists\big(true\ \mathsf{U}\ (\exists\Box\,\neg a)\big)\Big) = S \setminus \emptyset = S$

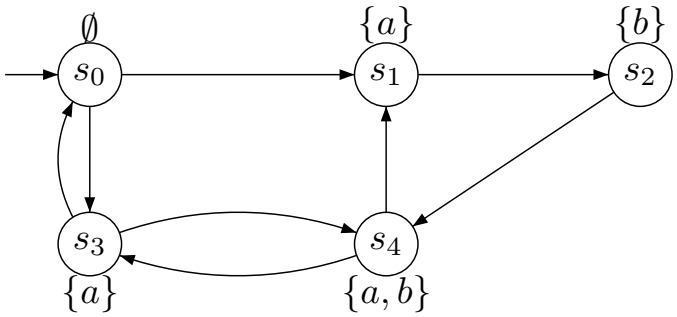
As $S_0 \subseteq S$, the formula Φ is satisfied in TS with respect to the fairness constraints $sfair$.

Exercise 3 (6 Points)

Consider the CTL-formula $\Phi = \forall\Box\,\big(a \rightarrow \forall\Diamond\,(b \wedge \neg a)\big)$ together with the following CTL fairness assumption

$$fair = \Box\Diamond\forall\bigcirc\,(a \wedge \neg b) \rightarrow \Box\Diamond\forall\bigcirc\,(b \wedge \neg a)$$
$$\wedge\,\Diamond\Box\exists\Diamond\,b \rightarrow \Box\Diamond\,b.$$

Check whether $TS \models_{fair} \Phi$ for the transition system TS below.



Solution:

First we check the state formulae of the fairness assumptions. We have:

- $Sat(\underbrace{\forall\bigcirc\,(a \wedge \neg b)}_{b_1}) = \{s_0, s_4\}$
- $Sat(\underbrace{\forall\bigcirc\,(b \wedge \neg a)}_{c_1}) = \{s_1\}$
- $Sat(\underbrace{\exists\Diamond\,b}_{b_2}) = S$
- $Sat(\underbrace{b}_{c_2}) = \{s_2, s_4\}$

Next we need to check for which states $s \in S$ the fairness assumption is realizable, that is $s \models_{fair} \exists\Box\, true$.

To show realizability it suffices to find a *strongly connected node set*[1] that is reachable from s and satisfies both constraints of *fair*.

Treatment of $\exists\Box$ under strong fairness CTLFAIR4.4-20A

$$fair = \bigwedge_{1 \le i \le k} (\Box\Diamond b_i \rightarrow \Box\Diamond c_i)$$

$s \models_{fair} \exists\Box a$ iff there exists a non-trivial strongly connected node-set D of G_a such that

(1) D is reachable from s

(2) for all $1 \le i \le k$:

$$D \cap Sat(b_i) = \varnothing \quad \text{or} \quad D \cap Sat(c_i) \neq \varnothing$$

G_a: digraph that arises from $\mathcal{T}$ by removing all states s' with $s' \not\models a$

152 / 132

While the statement from the lecture (and the corresponding algorithm) can only deal with strong fairness constraints, weak fairness can be integrated easily. For the weak fairness constraint $\Diamond\Box b_2 \rightarrow \Box\Diamond c_2$, instead of requiring

$$D \cap Sat(b_2) = \emptyset \text{ or } D \cap Sat(c_2) \neq \emptyset,$$

we need

$$D \not\subseteq Sat(b_2) \text{ or } D \cap Sat(c_2) \neq \emptyset.$$

We observe that the whole state space S is an SCC and therefore also a strongly connected node set. In particular, we have

$$S \cap Sat(c_1) \neq \emptyset \text{ and } S \cap Sat(c_2) \neq \emptyset.$$

Therefore, all states that can reach S, i.e. for all $s \in S$, it is $s \models_{fair} \exists\Box\, true$ and we attach the special atomic proposition a_{fair} to all $s \in S$.

We translate Φ into ENF as follows.

$$\begin{aligned}
\Phi &= \forall\Box\,(a \rightarrow \forall\Diamond\,(b \wedge \neg a)) \\
&\equiv \neg\exists\Diamond\,\neg(a \rightarrow \forall\Diamond\,(b \wedge \neg a)) \\
&\equiv \neg\exists\Diamond\,\neg(\neg a \vee \forall\Diamond\,(b \wedge \neg a)) \\
&\equiv \neg\exists\Diamond\,(a \wedge \neg\forall\Diamond\,(b \wedge \neg a)) \\
&\equiv \neg\exists\Diamond\,(a \wedge \exists\Box\,(\neg b \vee a)) \\
&\equiv \neg\exists\,(true\ \mathsf{U}\ a \wedge \exists\Box\,(\neg b \vee a)) \\
&= \Phi'
\end{aligned}$$

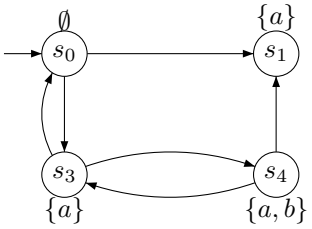
We now apply (fair) CTL model checking for Φ':

- $Sat_{fair}(a) = Sat(a) = \{s_1, s_3, s_4\}$

[1] Mind the difference to an SCC which has to be maximal, i.e. an SCC cannot be a subgraph of a bigger SCC.

- $Sat_{fair}(b) = Sat(b) = \{s_2, s_4\}$
- $Sat_{fair}(\neg b) = S \setminus Sat_{fair}(b) = \{s_0, s_1, s_3\}$
- $Sat_{fair}(\neg b \vee a) = Sat(\neg b \vee a) = \{s_0, s_1, s_3, s_4\}$
- $Sat_{fair}(\exists \Box (\neg b \vee a)) \ldots$

  For this subformula, we need to apply the *CheckFair* algorithm from the lecture on the subgraph $G_{\neg b \vee a}$ that only keeps states in $Sat(\neg b \vee a) = \{s_0, s_1, s_3, s_4\}$, which looks as follows:



Recursive algorithm *CheckFair*(...) CTLFAIR4.4-28

pseudo code for *CheckFair*$(C, k, \bigwedge_{1 \le i \le k} (\Box\Diamond b_i \rightarrow \Box\Diamond c_i))$

```
IF ∀i ∈ {1,...,k}. C ∩ Sat(c_i) ≠ ∅ THEN return "true" FI
choose j ∈ {1,...,k} with C ∩ Sat(c_j) = ∅;
remove all states in Sat(b_j);
IF the resulting graph G is acyclic THEN return "false" FI
FOR ALL nontrivial SCCs D of G DO
  IF CheckFair(D, k−1, ⋀_{i≠j} (□◇b_i → □◇c_i))
  THEN return "true"                          FI
OD
return "false"
```

215 / 132

$G_{\neg b \vee a}$ has only one SCC $C = \{s_0, s_3, s_4\}$ for which *CheckFair* returns *false*, because

- $C \cap Sat(c_1) = C \cap \{s_1\} = \emptyset$, and
- after removing the states in $Sat(b_1) = \{s_0, s_4\}$ the remaining graph is acyclic.

Intuitively, this answer also makes sense, because **no** state in C satisfies $c_1 = \forall \bigcirc (b \wedge \neg a)$ but it is impossible to avoid states that satisfy $b_1 = \forall \bigcirc (a \wedge \neg b)$ from some point on.

Hence $Sat_{fair}(\exists \Box (\neg b \vee a)) = \emptyset$

- $Sat_{fair}(a \wedge \exists \Box (\neg b \vee a)) = Sat(a \wedge \exists \Box (\neg b \vee a)) = \emptyset$
- $Sat_{fair}(\exists (true \, \mathsf{U} \, (a \wedge \exists \Box (\neg b \vee a)))) = Sat(\exists (true \, \mathsf{U} \, (a \wedge \exists \Box (\neg b \vee a) \wedge a_{fair})) = \emptyset$
- $Sat_{fair}(\Phi') = S$

We have that $Sat_{fair}(\Phi) = Sat_{fair}(\Phi') = S \supseteq I$ and therefore $TS \models_{fair} \Phi$.

Exercise 4 (5* + 3* Points)

***This exercise does not count towards the total number of points that you can achieve. Not solving it does not decrease the percentage of points you achieved while solving it may increase it.**

Consider the fragment ECTL of CTL which consists of formulae built according to the following grammar:

$$\Phi ::= a \mid \neg a \mid \Phi \wedge \Phi \mid \exists \varphi$$
$$\varphi ::= \bigcirc \Phi \mid \Box \Phi \mid \Phi \, \mathsf{U} \, \Phi$$

ECTL-formulae are built by atomic propositions, negated atomic propositions, the Boolean connective $\wedge$ and the path quantifier $\exists$ together with the modalities $\bigcirc, \Box$ and U. In particular, negation may only appear in front of atomic propositions.

For two transition systems $\mathrm{TS}_1 = (S_1, Act, \to_1, I_1, AP, L_1)$ and $\mathrm{TS}_2 = (S_2, Act, \to_2, I_2, AP, L_2)$, we define $\mathrm{TS}_1 \subseteq \mathrm{TS}_2$ iff $S_1 \subseteq S_2$, $\to_1 \subseteq \to_2$, $I_1 = I_2$ and $L_1(s) = L_2(s)$ for all $s \in S_1$.

(a) Prove by structural induction on the structure of the formula that for all ECTL-formulae Φ and all transition systems TS_1, TS_2 with $\mathrm{TS}_1 \subseteq \mathrm{TS}_2$, it holds:

$$\mathrm{TS}_1 \models \Phi \quad \Longrightarrow \quad \mathrm{TS}_2 \models \Phi.$$

Hint: You may gain one additional point by showing that the statement does not hold if the definition of $\mathrm{TS}_1 \subseteq \mathrm{TS}_2$ required only $I_1 \subseteq I_2$ instead of $I_1 = I_2$.

(b) Formally prove that there exists a CTL formula Φ_1 for which no equivalent ECTL formula Φ_2 exists.

Solution:

(a) We prove that for $\mathrm{TS}_1 \subseteq \mathrm{TS}_2$ and any arbitrary ECTL-formula Φ, we have

$$\mathrm{TS}_1 \models \Phi \quad \Longrightarrow \quad \mathrm{TS}_2 \models \Phi$$

We prove a slightly stronger claim, namely for all $s \in S_1$:

$$\mathrm{TS}_1, s \models \Phi \quad \Longrightarrow \quad \mathrm{TS}_2, s \models \Phi. \tag{10.1}$$

Since $I_1 = I_2$, we then have:

$$\begin{aligned}
&\mathrm{TS}_1 \models \Phi \\
&\iff \forall s \in I_1 \,.\, \mathrm{TS}_1, s \models \Phi \\
&\overset{10.1}{\Longrightarrow} \forall s \in I_1 \,.\, \mathrm{TS}_2, s \models \Phi \\
&\overset{I_1 = I_2}{\iff} \forall s \in I_2 \,.\, \mathrm{TS}_2, s \models \Phi \\
&\iff \mathrm{TS}_2 \models \Phi
\end{aligned}$$

In the following, for a path $\pi = s_0 s_1 \ldots$ we let $\pi[i]$ refer to s_i.

We prove (10.1) by induction on the syntactic structure of Φ:

- *Induction base:*
  - Let $\Phi = a \in AP$ and $\mathrm{TS}_1, s \models \Phi$. Then $a \in L_1(s)$. Because of $\mathrm{TS}_1 \subseteq \mathrm{TS}_2$, we have $s \in S_2$ and $L_1(s) = L_2(s)$. Therefore $a \in L_2(s)$ and $\mathrm{TS}_2, s \models a = \Phi$.

– Let $\Phi = \neg a$ for $a \in AP$ (Note that we only allow negations of atomic propositions!) and $\mathrm{TS}_1, s \models \neg a$. Then $a \notin L_1(s) = L_2(s)$ (since $s \in S_2$). Therefore also $\mathrm{TS}_2, s \models \neg a$.

- *Induction step:*
  – Let $\Phi = \Phi_1 \wedge \Phi_2$:

$$\begin{aligned}
&\mathrm{TS}_1, s \models \Phi \\
&\iff \mathrm{TS}_1, s \models \Phi_1 \text{ and } \mathrm{TS}_1, s \models \Phi_2 \\
&\overset{\text{I.H.}}{\Longrightarrow} \mathrm{TS}_2, s \models \Phi_1 \text{ and } \mathrm{TS}_2, s \models \Phi_2 \\
&\iff \mathrm{TS}_2, s \models \Phi
\end{aligned}$$

  – Let $\Phi = \exists \bigcirc \Psi$. We have

$$\begin{aligned}
&\mathrm{TS}_1, s \models \Phi \\
&\iff \mathrm{TS}_1, s \models \exists \bigcirc \Psi \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \pi \models \bigcirc \Psi \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \pi[1] \models \Psi \\
&\iff \exists (s, \alpha, s') \in \to_1 .\ \mathrm{TS}_1, s' \models \Psi \qquad (*\ \pi[1] = s'\ *) \\
&\overset{\to_1 \subseteq \to_2}{\Longrightarrow} \exists (s, \alpha, s') \in \to_2 \ .\ s' \in S_2 \text{ and } \mathrm{TS}_1, s' \models \Psi \\
&\overset{\text{I.H.}}{\Longrightarrow} \exists (s, \alpha, s') \in \to_2 .\ \mathrm{TS}_2, s' \models \Psi \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_2}(s).\ \pi[1] \models \Psi \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_2}(s).\ \pi \models \bigcirc \Psi \\
&\iff \mathrm{TS}_2, s \models \exists \bigcirc \Psi \\
&\iff \mathrm{TS}_2, s \models \Phi
\end{aligned}$$

  – Let $\Phi = \exists \Box \Psi$. We have

$$\begin{aligned}
&\mathrm{TS}_1, s \models \Phi \\
&\iff \mathrm{TS}_1, s \models \exists \Box \Psi \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \pi \models \Box \Psi \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \forall i \geq 0.\ \mathrm{TS}_1, \pi[i] \models \Psi
\end{aligned}$$

  Because of $S_1 \subseteq S_2$, $\pi[i] \in S_2$ for all $i \geq 0$. By induction hypothesis, we have for all $i \geq 0$:

$$\mathrm{TS}_1, \pi[i] \models \Psi \quad \overset{\text{I.H.}}{\Longrightarrow} \quad \mathrm{TS}_2, \pi[i] \models \Psi.$$

  $\pi \in \mathit{Paths}_{\mathrm{TS}_1}(s)$, so for all $i \geq 0$ it is $\pi[i] \overset{\alpha_i}{\to}_1 \pi[i+1]$ for some $\alpha_i \in \mathrm{Act}$. Since $\to_1 \subseteq \to_2$, we have $\pi[i] \overset{\alpha_i}{\to}_2 \pi[i+1]$ for all $i \geq 0$ and therefore $\pi \in \mathit{Paths}_{\mathrm{TS}_2}(s)$. Therefore we have

$$\begin{aligned}
&\exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \forall i \geq 0.\ \mathrm{TS}_1, \pi[i] \models \Psi \\
&\Longrightarrow \exists \pi \in \mathit{Paths}_{\mathrm{TS}_2}(s).\ \forall i \geq 0.\ \mathrm{TS}_2, \pi[i] \models \Psi \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_2}(s).\ \pi \models \Box \Psi \\
&\iff \mathrm{TS}_2, s \models \exists \Box \Psi \\
&\iff \mathrm{TS}_2, s \models \Phi
\end{aligned}$$

  – Let $\Phi = \exists \Phi_1 \,\mathsf{U}\, \Phi_2$: Here, we have

$$\begin{aligned}
&\mathrm{TS}_1, s \models \exists \Phi_1 \,\mathsf{U}\, \Phi_2 \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \pi \models \Phi_1 \,\mathsf{U}\, \Phi_2 \\
&\iff \exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \exists j \geq 0.\ \pi[j] \models \Phi_2 \wedge \forall i < j.\ \pi[i] \models \Phi_1
\end{aligned}$$

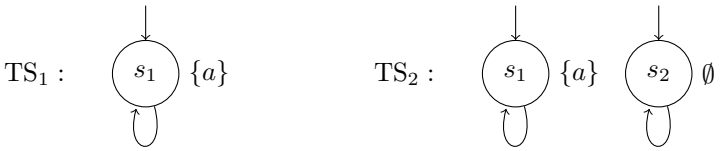
From $S_1 \subseteq S_2$ it follows that $\pi[j] \in S_2$ and that $\pi[i] \in S_2$ for all $i \in \{0, \ldots, j-1\}$. Applying the induction hypothesis yields:

$$\begin{aligned} &\mathrm{TS}_1, \pi[j] \models \Phi_2 && \Longrightarrow \mathrm{TS}_2, \pi[j] \models \Phi_2 \\ &\mathrm{TS}_1, \pi[i] \models \Phi_1 && \Longrightarrow \mathrm{TS}_2, \pi[i] \models \Phi_1 \quad \text{for all } i \in \{0, \ldots, j-1\} \end{aligned}$$

As $\rightarrow_1 \subseteq \rightarrow_2$ we have $\pi \in \mathit{Paths}_{\mathrm{TS}_2}(s)$ as shown before. Therefore

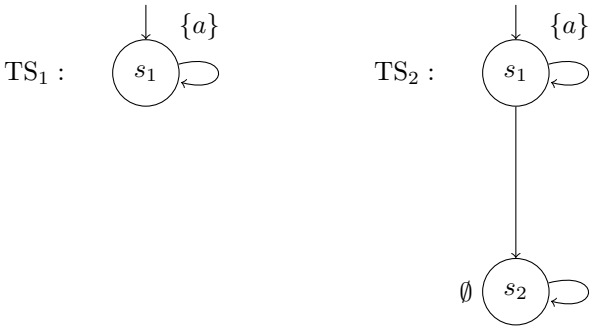
$$\begin{aligned} &\exists \pi \in \mathit{Paths}_{\mathrm{TS}_1}(s).\ \exists j \geq 0.\ \pi[j] \models \Phi_2 \wedge \forall i < j.\ \pi[i] \models \Phi_1 \\ \Longrightarrow\ &\exists \pi \in \mathit{Paths}_{\mathrm{TS}_2}(s).\ \exists j \geq 0.\ \pi[j] \models \Phi_2 \wedge \forall i < j.\ \pi[i] \models \Phi_1 \\ \Longleftrightarrow\ &\exists \pi \in \mathit{Paths}_{\mathrm{TS}_2}(s).\ \pi \models \Phi_1 \,\mathsf{U}\, \Phi_2 \\ \Longleftrightarrow\ &\mathrm{TS}_2, s \models \exists \Phi_1 \,\mathsf{U}\, \Phi_2 \end{aligned}$$

Finally, we illustrate that the claim does not hold if we require $I_1 \subseteq I_2$ instead of $I_1 = I_2$. Consider the ECTL formula $\Phi = \exists \Box\, a$ and the two transition systems TS_1 and TS_2 below.



Except for $I_1 = I_2$, we have $\mathrm{TS}_1 \subseteq \mathrm{TS}_2$. We do have $I_1 \subseteq I_2$. We observe $\mathrm{TS}_1 \models \Phi$. However, in TS_2 there is an initial state, namely s_2 with $s_2 \not\models \Phi$ and therefore $\mathrm{TS}_2 \not\models \Phi$, which is a contradiction to the claim.

(b) We prove that the formula $\Phi_1 = \forall \Box\, a$ is not equivalent to any ECTL formula Φ_2. For the sake of contradiction, assume that there is Φ_2 in ECTL such that $\Phi_1 \equiv \Phi_2$. Consider the following transition systems TS_1 and TS_2.



It is $\mathrm{TS}_1 \subseteq \mathrm{TS}_2$ and $\mathrm{TS}_1 \models \Phi_1$ and therefore $\mathrm{TS}_2 \models \Phi_2$ since $\Phi_1 \equiv \Phi_2$. By the theorem that we proved, we have that $\mathrm{TS}_1 \models \Phi_2 \implies \mathrm{TS}_2 \models \Phi_2$ and therefore $\mathrm{TS}_2 \models \Phi_1$ since $\Phi_1 \equiv \Phi_2$. However, we find that $\mathrm{TS}_2 \not\models \Phi$, contradiction.