Principles Of Model Checking Exercises Solutions

Dario Bekic

October 6, 2025

1 Introduction

These are my solutions for the exercises of the book Principles Of Model Checking by Christel Baier and Joost-Pieter Katoen

2 3 Linear Time Properties

2.1 Exercise 3.1

This is like giving a regular grammar.

$$
\begin{aligned}
&S \to aS' | \emptyset S'' \\
&S' \to aS''' | b''' | \\
&S'' \to aS''' | b''' \\
&S''' \to aS'
\end{aligned}
$$

Exercise 3.2

a) $f : TS \mapsto TS^*$ (set TS of transitions systems with terminal states and TS^* is the one without). $f(S, Act, \to, I, AP, L, F) = (S \sqcup \{s_f\}, Act, \to \cup \{(t, \tau, s_f) | t \in F\}, I \setminus F, AP, L \cup \{(s_f, \emptyset)\})$ by viewing everything as a set.

b) Suppose $\sigma \in Traces(TS_1^*) \wedge \sigma \notin Traces(TS_2^*)$. Let π_1 be a TS_1^* path such that $Trace(\pi_1) = \sigma$. Either $\exists q.\pi_q = s_f$ or not. Assume the first case, and let j be the index of the first element in π_1 such that $\pi_1 \neq s_f$. The path fragment $\pi_1' = \pi_{1,1}, ..., \pi_{1,j}$ is a maximal path in TS_1 since $\pi_{1,j}$ is a terminal state by construction. By hypothesis there must exist a path $\pi_2' \in TS_2$ such that $Trace(\pi_1') = Trace(\pi_2')$ thus also π_2' must be a maximal initial finite fragment path i.e. must end in a terminal state of TS_2. Now we can extend the path π_2' with infinite loops and since s_{fin} i.e. the phantom state added to TS_2^* has no atomic proposition its trace will be equal to that of π_1. Namely, $\sigma = Trace(\pi_1) = Trace(\pi_1') = Trace(\pi_2') = Trace(\pi_2)$, contradicting the claim that $\sigma \notin Traces(TS_2^*)$.

Exercise 3.3

Algorithm 1 BFS for Invariant Checking (shortest counterexample)

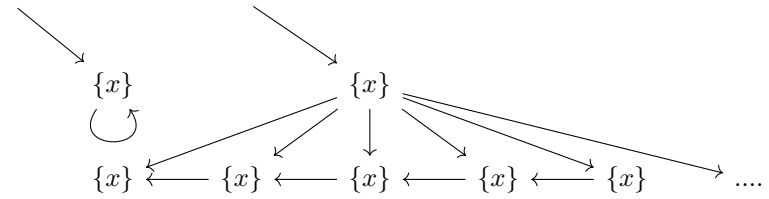
```
1: procedure BFSInvariant(TS(S, Act, →, I, AP, L), φ)
2:     Q ← emptyQueue()
3:     pred ← emptyMap() ▷ pred[s] is predecessor of s; absent means unseen
4:     for all s ∈ I do
5:         if L(s) ⊭ φ then
6:             return ⟨false, [s]⟩
7:         end if
8:         Q.enqueue(s)
9:         pred[s] ← ⊥                    ▷ mark as seen; ⊥ denotes start of path
10:    end for
11:    while ¬Q.empty() do
12:        u ← Q.dequeue()
13:        for all v ∈ post(u) do
14:            if v ∉ keys(pred) then              ▷ first time seen ⇒ shortest
15:                pred[v] ← u
16:                if L(v) ⊭ φ then
17:                    return ⟨false, Reconstruct(pred, v)⟩
18:                end if
19:                Q.enqueue(v)
20:            end if
21:        end for
22:    end while
23:    return ⟨true, emptyList⟩          ▷ invariant holds on all reachable states
24: end procedure
25: procedure Reconstruct(pred, t)
26:    path ← emptyList
27:    x ← t
28:    while x ≠ ⊥ do
29:        path.append(x)
30:        x ← pred[x]
31:    end while
32:    return reverse(path)
33: end procedure
```

Exercise 3.4

a) $\Longrightarrow$: pick a finite path fragment $\pi \in PathFragment_{fin}(TS)$ such that $Trace(\pi) \in Traces_{fin}(TS) \wedge Trace(\pi) \notin Traces_{fin}(TS')$. Call σ the size of π. I can extend π to a maximal path π' such that $Trace(\pi') \in Traces(TS) \wedge Trace(\pi') \in Traces(TS')$. By $Trace(\pi') \in Traces(TS')$ there must exist a path $\rho' \in Paths(TS')$ such that $Trace(\rho') = Trace(\pi')$.

Thus, the prefix of length σ of ρ' is a finite path fragment ρ in TS' such that $Trace(\rho) = Trace(\pi)$, contradicting the hypothesis $Traces(\pi) \notin Traces_{fin}(TS')$. The same can be argued by swapping TS and TS' roles, completing the proof.
$\Longleftarrow$: pick a path $\xi \in Paths(TS)$ such that $Trace(\xi) \in Traces(TS) \wedge Trace(\xi) \notin Traces(TS')$. I argue that for any $n \in \mathbb{N}$ there exists a path fragment p^n in TS' such that its trace its equal to the prefix of length n of ξ and that it can be extended to path fragment of size $n+1$ which is equal to the trace of the prefix of length $n+1$ of ξ. The first part of the statement comes directly from the assumption that the finite traces sets equal each other. The second part comes from the fact any trace in both transition systems can be given by only one path (or path fragment), thus for the prefix of length $n+1$ of ξ i need necessarily to extend p^n because if there was another path different from p^n, call it p^*, then their prefix trace of size n would be the same, contradicting the claim. Suppose that in fact such p^* existed, then it may share a prefix of states with p^n, let $q < n$ be the last state they share, i.e. $p^*_q = p^n_q$ then there should be two states p^*_{q+1} and p^n_{q+1} such that $L(p^*_{q+1}) = L(p^n_{q+1})$ because p^*, p^n share the same trace up to the first n states, but this contradicts AP determinism.

b)



Exercise 3.5

a) $\texttt{false} \triangleq \emptyset$

b) $\texttt{initEqualZero} \triangleq \{A_0A_1, ... \in (2^{AP})^\omega | \exists i \in \mathbb{N}.A_i = \{x = 0\} \wedge \forall j < i.A_j = \emptyset\}$

c) $\texttt{initDiffZero} \triangleq (2^{AP})^\omega \setminus \texttt{initEqualZero}$

d) This seems not to be safety nor liveness.

e) Liveness property

f) Liveness property

g) $\texttt{initEqualZero} \triangleq \{A_0A_1, ... \in (2^{AP})^\omega | (A_i = \{x = 0\} \wedge A_j = \{x > 0\} \wedge$ i even, j odd$) \vee (A_i = \{x = 0\} \wedge A_j = \{x > 0\} \wedge$ i odd, j even$)\}$

h) $\texttt{true} \triangleq (2^{AP})^\omega$

Exercise 3.6

a) Invariant: `Never` $\triangleq \{A_0A_1, ... \in (2^{AP})^\omega | A \notin A_i \forall i\}$

b) Safety: `Once` $\triangleq \{A_0A_1, ... \in (2^{AP})^\omega | \exists i.A \in A_i \wedge \forall j.(j < i \vee j > i) \wedge A \notin A_j\}$

c) Liveness: `AlternateInfinitely` $\triangleq \{A_0A_1, ... \in (2^{AP})^\omega | \exists i, j, q.j > i \wedge A_j = \{A\} \wedge q > i \wedge A_q = \{B\} \wedge (\forall t > i.A_t = \{A\} \implies \exists x.A_x = \{B\} \wedge \forall c.i < c < x.B \notin A_c) \wedge (\forall t > i.A_t = \{B\} \implies \exists x.A_x = \{A\} \wedge \forall c.i < c < x.A \notin A_c)\}$ (its a liveness property because the i existential witness can be large as one wishes).

d) Liveness `AlternateInfinitely` $\triangleq \{A_0A_1, ... \in (2^{AP})^\omega | \exists i.((\exists j < i.A \in A_j \wedge \forall c.j < c < i.B \notin A_c) \implies B \in i) \wedge (\forall q.q > i \wedge (A \in A_q \implies \exists t.t > q.B \in A_t))\}$

Exercise 3.7

Initial values: $r_1 = 0, r_2 = 1$. y value is given by the value of register r_2 and the formula is $((r_1 \oplus x) \vee (x \wedge r_1))$. r_1 value is given by $(r_1 \oplus x) \vee (r_2 \wedge x)$

a) Let α_{P_i} be a word in P_i and β_{P_i} be a word in $(2^{AP})^\omega \backslash P_i$. $\alpha_{P_1} \triangleq \{r_2, x, y\}(A)^\omega$ where $A \triangleq AP$, $\alpha_{P_2} \triangleq \{r_1, x\} \cdot (\emptyset)^\omega$, $\alpha_{P_3} \triangleq (\{y\}, \emptyset)^\omega$, $\alpha_{P_4} \triangleq \emptyset^\omega$, $\beta_{P_1} \triangleq \{x\}^\omega$, $\beta_{P_2} \triangleq \{r_2, r_1\}^\omega$, $\beta_{P_3} \triangleq \{y\}^\omega$, $\beta_{P_4} \triangleq \{x, r_1\}^\omega$

b) P1) If $x = 1$ is high will be high regardless of r_1: if $r_1 = 0$, $r_1 \oplus x$ will be 1, if $r_1 = 1$ then $x \wedge r_1$ will be 1. Thus it is satisfied.

P2) If $r_2 = 0, r_1 = 0$ in any state, in the next state we will $r_1 = 1$ if $x = 1$ so it is not satisfied.

P3) This is not satisfied because P_1 is satisfied.

P4) False, it is a possible initial state

c) P2, P3, P4 are safety properties and only P4 is an invariant.

i) Let X_i be equal to any set in $\{x, r_1, r_2, y\}$ of size i. S^* is the terminal state of the grammar: with that symbol we can expand to any set of properties.

P2)

$$S \to X_i \setminus \{r_2\}S' | X_i \cup \{r_2\}S$$
$$S' \to X_iS' | X_i \cup \{r_1\}S^*$$

P3)

$$S \to X_iS | X_i \cup \{y\}S'$$
$$S' \to X_i \cup \{y\}S*$$

P4)

$$S \to X_i S | X_i \cup \{x\} \setminus \{r_1\} S^*$$

ii) There is only $P4$: $\phi \triangleq \neg x \vee r_1$

Exercise 3.8

$\Longrightarrow$: this direction holds and directly comes from the definition of closure of a property: $closure(P) = \{\alpha \in (2^{AP})^\omega | pref(\alpha) \subseteq pref(P) = \{\alpha \in (2^{AP})^\omega | pref(\alpha) \subseteq pref(P') = closure(P')\}$.
$\Longleftarrow$: suppose $closure(P) = closure(P')$ but $\alpha \in pref(P) \wedge \alpha \notin pref(P')$. Since $\alpha \in pref(P)$ there exists $\sigma \in P$ such that α is a prefix of σ. By definition of closure, $\sigma \in closure(P)$ which implies $\sigma \in closure(P')$ by hypothesis. But $\sigma \in closure(P') \iff pref(\sigma) \subseteq pref(P')$ by definition of closure, and thus we have $\alpha \in pref(P')$.

Exercise 3.9

We need to show that i) the set $\Phi \triangleq closure(Traces(TS))$ is a LT safety property and ii) that TS satisfies it.

i) imagine there were $\alpha = A_0 A_1, ... \in (2^{AP})^\omega \setminus \Phi$ such that there exists $\pi \in pref(\alpha)$ such that $\Phi \cap \{\beta \in (2^{AP})^\omega | \pi \in pref(\beta)\} \neq \emptyset$. Since $\alpha \notin closure(Traces(TS))$ it means there is a finite prefix of α that is not a prefix of some trace in TS. Let this trace be our π. Then if $\sigma \in \Phi \cap \{\beta \in (2^{AP})^\omega | \pi \in pref(\beta)\}$ it means $pref(\sigma) \subseteq pref(Traces(TS))$ which means $\pi \in pref(Traces(TS))$.

ii) A word ξ is in Φ if and only if $pref(\xi) \subseteq pref(Traces(TS))$. Any path ρ in TS has $Trace(\rho) \in Traces(TS)$ thus $pref(Trace(\rho)) \subseteq pref(Traces(TS))$.

Exercise 3.10

Suppose $\phi \in pref(closure(P))$ then $\exists \rho \in closure(P)$ such that a finite prefix of ρ is ϕ. But any element in $closure(P)$ is just a word whose prefixes are elements of $pref(P)$ thus any prefix of ρ is a an element of $pref(P)$, even ϕ, thus $\phi \in pref(P)$.

Exercise 3.11

a) $\texttt{UNION\_SAFE} \triangleq P \cup P'$ is a safety property if P, P' are safety properties. Consider $w \in \texttt{UNION\_SAFE}$. For it to be a safety property, it is sufficient to prove that for any $\rho \in (2^{AP})^\omega \setminus (\texttt{UNION\_SAFE})$ and for any finite prefix π of it $(\texttt{UNION\_SAFE}) \cap (\underbrace{\{\alpha \in (2^{AP})^\omega | \pi \text{ is a finite prefix of } \alpha\}}_{\triangleq \texttt{PREFIX}}) = \emptyset$. This equals to say, by distributivity, $(P \cap \texttt{PREFIX}) \cup (P' \cap \texttt{PREFIX}) = \emptyset$. Since P, P' are safety properties we have $\emptyset \cup \emptyset = \emptyset$.

b) $\texttt{INTER\_SAFE} \triangleq P \cap P'$ is a safety property if P, P' are safety properties. Take any word $w \in (2^{AP})^\omega$. There are 2 cases to consider:

   i) $w \notin P \cup P'$: we wish to show that for any finite prefix $\pi \in pref(w)$, $P \cap P' \cap \underbrace{\{\alpha \in (2^{AP})^\omega | \pi \text{ is a finite prefix of } \alpha\}}_{\triangleq \texttt{PREFIX}}$. But since P' is a safety property $P' \cap \texttt{PREFIX} = \emptyset$, thus $P \cap \emptyset = \emptyset$.

   ii) $w \in P' \wedge w \notin P$: we wish to show again that for any finite prefix $\pi \in pref(w)$, $\texttt{INTER\_SAFE} \cap \texttt{PREFIX} = \emptyset$. By commutativity of intersection, $P \cap P' \cap \texttt{PREFIX} = P' \cap P \cap \texttt{PREFIX} = P' \cap \emptyset = \emptyset$. The case of $w \in P \wedge w \notin P'$ is identical.

c) $\texttt{UNION\_LIVE} \triangleq P \cup P'$ is a liveness property if P, P' are liveness properties. This is trivially arguable informally: if i can extend any finite prefix to a trace in $w_p \in P$ or to a trace $w'_p \in P'$ then i can extend any prefix to a trace in $P \cup P'$ (just pick either w_p or w'_p).

d) $\texttt{INTER\_LIVE} \triangleq P \cap P'$ is not necessarily a liveness property. To see a concrete counter-example consider $P \triangleq \{$ from one point there will be only 1s$\}$ and $P' \triangleq \{$ from one point there will be only 0s$\}$. Their intersection is empty, and the empty set is not a liveness property.

Exercise 3.12

1. $closure(P) \subseteq P_{safe}$. Assume there exists $\alpha \in closure(P)$ such that $\alpha \notin P_{safe}$. Since $\alpha \in closure(P)$, for any finite prefix β of α, $\beta \in pref(\alpha) \implies \beta \in pref(P)$. So pick any prefix β of α, since P_{safe} is a safety property we have $P_{safe} \cap \{\gamma | \beta \in pref(\gamma)\} = \emptyset$. But we also proved $\beta \in pref(P)$, and since P contains a subset of P_{safe} it means $\exists \lambda \in P_{safe}.\beta \in pref(\lambda)$. But this would imply $\lambda \in P_{safe} \cap \{\gamma | \beta \in pref(\gamma)\} \neq \emptyset$.

2. $P_{live} \subseteq \underbrace{P \cup ((2^{AP})^\omega \setminus closure(P))}_{\texttt{EXPR}}$. Suppose $w \in P_{live}$ but not in $\texttt{EXPR}$. This can only happen if $w \notin P$ which implies $w \notin P_{safe}$. We wish to prove $w \in \texttt{EXPR}$ if $w \notin P_{safe}$. This is equal to proving $w \notin P_{safe} \implies w \notin closure(P)$ by definition of $\texttt{EXPR}$. We can prove the contrapositive: $w \in closure(P) \implies w \in P_{safe}$ which we proved in the first point.

Exercise 3.13

One can of course use the Decomposition theorem, but there is a way of giving explicitly the two properties by using intuition.

One would of course first start from characterizing P: maybe the exercise is tricky and P is a safe or liveness property. Proving one such result would let us quickly end the exercise.

a. Is P a safety property? One could argue that the set of bad prefixes is $\{w_1, w_2, .. \in (2^{AP})^\omega | \exists n \in \mathbb{N}. w_n = \{a, b\} \wedge \exists j \in \mathbb{N}. j < n \wedge a \notin w_j\}$. However, the condition to have *infinitely* many $j \in \mathbb{N}. b \in w_j$ cannot be captured by a finite prefix. In conclusion, P is not a safety property.

b. Is P a liveness property? No, clearly $w' = \{\{b\}, \{a, b\}\}$ cannot be extended to a word satisfying P.

Even though we were unsuccessful in proving safety or liveness, we identified the parts that were problematic during our proof attempt.

In particular:

$$P_{safe} \triangleq (2^{AP})^\omega \setminus \{w_1, w_2, .. \in (2^{AP})^\omega | \exists n \in \mathbb{N}. (w_n = \{a, b\} \wedge \exists j \in \mathbb{N}. j < n \wedge a \notin w_j)\}$$

and

$$P_{live} \triangleq \{w_1, w_2, ... | \overset{\infty}{\exists} j. b \in w_j\}$$

The correctness is immediate.

Exercise 3.14

First, let's put here the transition graphs. Consider the TS for Peterson's algorithm in Figure 15 and for the Semaphore-based mutual exclusion in Figure 16. Note that we have req_1, req_2 labels even though they are not in AP just for ease of read. In general we can solve this exercise by reasoning on the LT properties that the transition systems satisfy or not. As we know, the semaphore transition systems does not guarantee starvation freedom i.e. it is possible that one process will never enter its critical region.

Thus, the trace:

$$w \triangleq \{\{req_1, req_2\}, \{wait_1, req_2, \}, \{crit_1, req_2\}\}^\omega$$

is in $Traces(TS_{sem})$ but not in $Traces(TS_{pet})$.

Exercise 3.15

(a) Regarding satisfaction of E': without *unconditional* fairness assumptions it is a matter of verifying if there exists a path which contains $\{\{a\}, \{a, b\}, \emptyset\}$ and this is true $s_1 \to s_2 \to s_3 \to s_1$, thus this property is never satisfied. For B_1 only E_a because we may loop on $s_1 \to s_2 \to s_1$. For B_2 both E_a and E_b as we need to exit the loop we mentioned because $s_1 \to s_3$ is infinitely enabled. For B_3 only E_b, because of the loops $s_1 \to s_2 \to s_1$ and $s_1 \to s_2 \to s_4 \to s_1$ we always pass by s_1 which has enabled the β transition to s_3.

(b) The same reasoning for E' applies. Now, recall that weaker assumptions restrict *the same* or more traces, thus we do not check again those we proved are not satisfied by strong assumptions. For example, B_1 now

looses E_a as α-transitions from s_1 are not continuously enabled, as we can loop on $s_1 \to s_3 \to s_1$. The same goes for B_3. And we also loose both E_a, E_b for B_2 as we can loop on $s_1 \to s_2 \to s_1$ continuously *alternating* between having α and β transitions.

Exercise 3.16

(a)
1. With no assumption this is false. Let $AP_1 = \{\alpha\}$ and $AP_2 = \{\beta\}$. Then let $\forall s \in S_2.\beta \in L(s), \forall s' \in S_1.\alpha \in L(s')$. Then $\forall A_1, A_2... \in Traces(TS_1||TS_2)$ we have $\beta \in A_i$ but no trace in TS_1 contains such label (as it is not part of TS_1 label set).
2. This is false. Consider Figure 1 where $\langle s_1, s'_1 \rangle$ is the initial state and consider $Syn = \{\alpha\}$. It will be impossible to move out of the initial state as s'_1 has no outgoing α transition to synchronize on. We have only have one trace: $Traces(TS_1||TS_2) = \{\{Trace(\langle s_1, s'_1 \rangle)\}\} = \{\{L_1(s_1)\}\}$

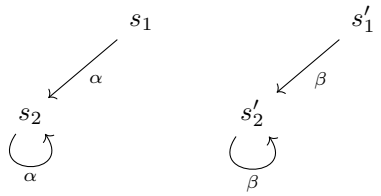


Figure 1: Exercise 3.16. point (a)

(b)
1. With no assumption it is the same as (a).
2. Assuming $AP_2 = \emptyset$ the result is true. Let us restrict to the subset of paths in $TS_1||TS_2$ that use only a transition from TS_1. We argue that for any path $\sigma_1 = s_0, s_1, ... \in Paths(TS_1)$ we have a path $\sigma_2 = \langle s'_0, s'_1, ... \rangle$ such that $\pi_1(s'_i) = s_i$, where π_1 is the left projection. This can be proved easily by induction.

(c)
1. With no assumption it is the same as (a).
2. Assuming $AP = \emptyset$ it is still false. Consider Figure 2. The transition system $TS_1||TS_2$ allows for the path $P \triangleq (\langle s_1, s'_1 \rangle \xrightarrow{\beta} \langle s_1, s'_1 \rangle)^\omega$ with $Trace(P) = a^\omega$. However this trace is not available to TS_1 whose trace set is $\{\{a \cdot b^\omega\}\}$.

(d) False. Consider Figure 3 and the two transition systems, where we just write the Atomic Propositions that hold: and call the TS on the left TS_1 and the one on the right TS_2. It is obvious that $Traces(TS_1) \subseteq Traces(TS_2)$ as $Traces(TS_1) = Traces(TS_2)$. Clearly for $\mathcal{F}_{strong} = \{\{\alpha\}\}$ TS_2 has no infinite fair traces as it *has* to take the α transition while TS_1 does.

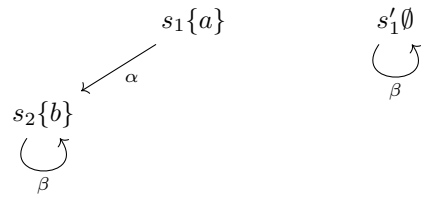


Figure 2: Exercise 3.16. point (c)

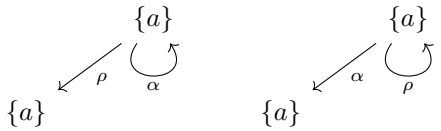


Figure 3: Exercise 3.16. point (d)

(e) We simply make a little change to the transitions systems of the point (d). As you can see in figure 4., we have clearly $Traces(TS_1) = \{a^\omega\} \cup \{a^n \cdot b | n \in \mathbb{N}\} = Traces(TS_2)$ thus $Traces(TS_1) \subseteq Traces(TS_2)$. Consider the liveness property:

$$\texttt{EVENTUALLY\_B} \triangleq \exists n.b \in A_n$$

Under $\mathcal{F} = \{\emptyset, \{\{\alpha\}\}, \emptyset\}$ we have $TS_2 \models_{\mathcal{F}}$ `EVENTUALLY_B` but $TS_1 \not\models_{\mathcal{F}}$ `EVENTUALLY_B`.

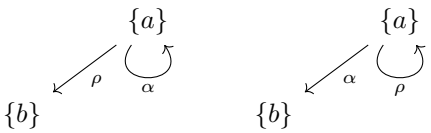


Figure 4: Exercise 3.16. point (e)

Exercise 3.17

It is enough to count all the cycles and see if one of those two non-mutually exclusive (but sufficient) conditions hold: *i*) to access the cycle by using any path we encounter a or *ii*) in at least one state of the cycle we have available either a α or β as outgoing transitions to states where a holds (or where we have proved that it eventually holds).
(Some) cycles are:

1 $s_1 \to s_1$: since in s_1 a holds we conclude by criterion *i*).

2 $s_2 \to s_3 \to s_2$: we have infinitely many times β transition available which sends us to s_1, thus by criterion *ii*) and point 1. we conclude.

3 $s_3 \to s_4 \to s_5 \to s_4 \to s_6 \to s_3$ both criterion do not hold, thus the property is not satisfied.

Exercise 3.18

(a) It is not realizable as there is no fair path starting from s_1 which is reachable with an α transition from s_0 because of the unconditional assumption $\{\alpha\}$.

(b) By having now the unconditional assumption $\{\delta, \alpha\}$ the loop $s_1 \to s_4 \to s_1$ is a fair path. Thus, $\mathcal{F}_2$ is realizable.

(c) This is too realizable. The loop $s_1 \to s_4 \to s_1$ will enable infinitely many times β and by strong fairness $\{\alpha, \beta\}$ we will be forced to take it infinitely many times thus satisfying the unconditional assumption $\{\beta\}$.

Exercise 3.19

(a) (i)

$$
\begin{aligned}
E &::= R_1(R_2^\omega) \\
R_1 &::= R_1' | \epsilon \\
R_1' &::= \emptyset R_1' | a \\
R_2 &::= \{a\}R_2' | \{b\}R_2 | \{a,b\}R_2 \\
R_2' &::= \{b\}R_2
\end{aligned}
$$

(ii) By the decomposition theorem we have

$$
\begin{aligned}
P_{safe} &= \{A_0A_1... \in (2^{AP})^\omega | ...\} \\
... &= \exists n.(A_n = \{a\} \wedge \forall j.j < n \wedge A_j = \emptyset \wedge \forall k.k > n.A_k = \{a\} \implies A_{k+1} = \{b\}) \cup \emptyset^\omega \\
P_{live} &= P_1 \cup ((2^{AP})^\omega \setminus closure(P)) \\
&= (\{b\} + \{a,b\})^\omega \cup P_1 \cup \{A_0A_1... \in (2^{AP})^\omega | ...\} \cup \{\{a\}A_1...|*\} \\
... &= \exists n.(n > 0 \wedge A_n = \{a\} \wedge \exists j < n.A_j \in \{\{a,b\},\{b\}\}) \\
* &= \exists i > 0.A_i = \{a\} \wedge A_{i+1} \neq \{b\}
\end{aligned}
$$

(iii)
1. Suppose $w \in (2^{AP})^\omega \setminus P_{safe}$. If $\nexists i.w_i = \{a\}$ then pick j the first non-empty set of w: it should be clear that any word with this prefix is not in P_{safe}. If $\exists i.w_i = \{a\}$ then it must be:
   i. Either $existsj.j < n \wedge A_j \neq \emptyset$. Then it is sufficient to pick the prefix $A_0...A_i$.
   ii. Or(they are not mutually exclusive) $\exists k > i.A_k = \{a\} \wedge A_{k+1} \neq b$. Pick the prefix $A_0, ...A_{k+1}$.

(b) First, let's re-draw the transition system which we can see in Figure 5, where each node is denoted with the pair $s\{L(s)\}$. we work separately for

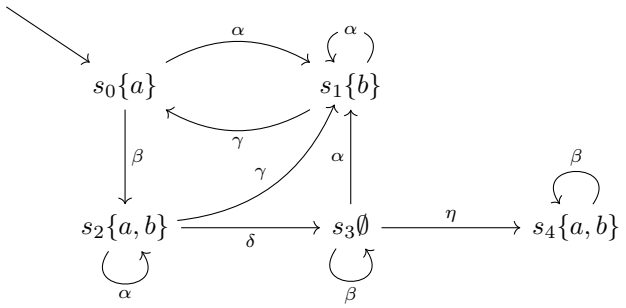


Figure 5: Exercise 3.19. point (b)

the two fairness assumptions. The first one is:

$$\mathcal{F}_1 = (\{\{\alpha\}\}, \{\{\beta\}, \{\gamma, \delta\}, \{\eta\}\}, \emptyset\})$$

First, state s_4 is ruled out as it will not allow for infinite α transitions that is a condition of the unconditional assumptions.

Notice that we will be forced to take the β transition from s_0 to s_2 as it is one of our strong assumptions forcing us out from the hypothetic9s

3 4 Regular Properties

3.0.1 Exercise 4.1

(a) yes, see Figure 6

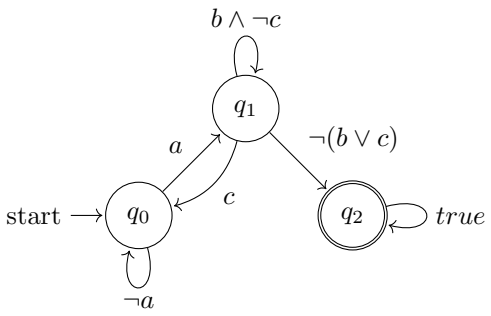


Figure 6: Exercise 4.1 point (a)

(b) yes, see Figure 7

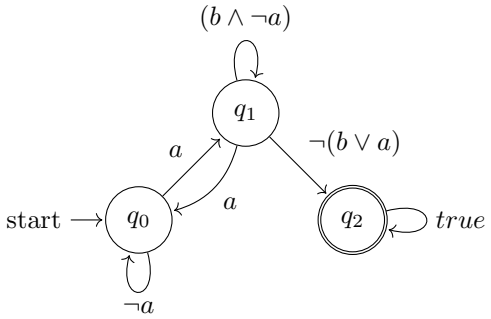


Figure 7: Exercise 4.1 point (b)

(c) No, as $L = c^n b^{n+1}, n \geq 0$ is a subset of the property but is not a regular language. To prove it one can use the Pumping Lemma: let P be the pumping length and let $w = c^L b^{L+1}$, clearly $c^{L+x} b^{L+1}$for $x > 0$ is not in the language.

(d) Yes, see Figure 8

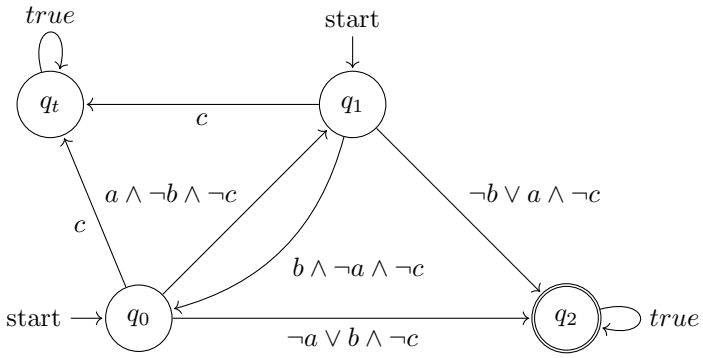


Figure 8: Exercise 4.1 point (b)

3.0.2 Exercise 4.2

1. Let $n = 3$ for simplicity.

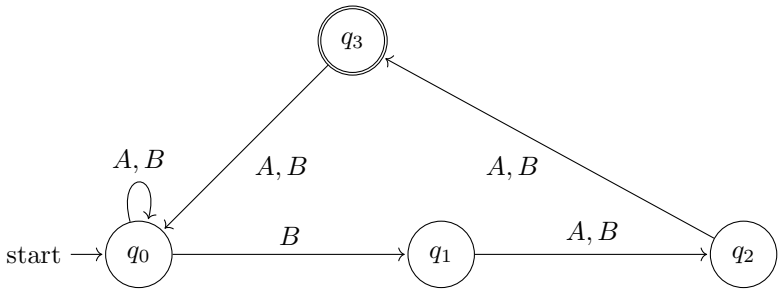


Figure 9: Exercise 4.2 point (a)

2. To determinize the NFA we start from q_0 and with a A transition we can only stay in q_0 while using B we can either move to q_0 or q_1, we call this state $q_{0,1}$.Then from this state by either A or B we go in the state $q_{0,2}$. Then, again by either A or B, we go to state $q_{0,3}$ and then again to state q_0. See Figure 10.

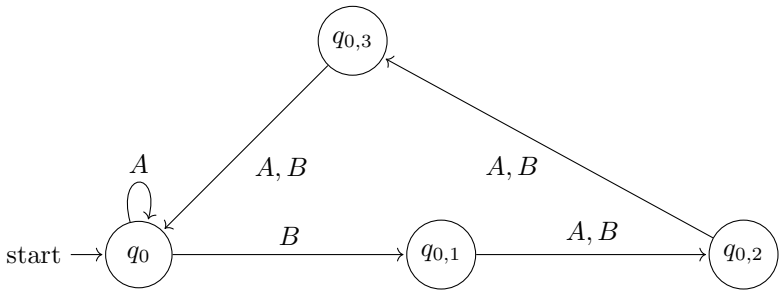


Figure 10: Exercise 4.2 point (b)

3.0.3 Exercise 4.3

(a) (i) See Figure 11

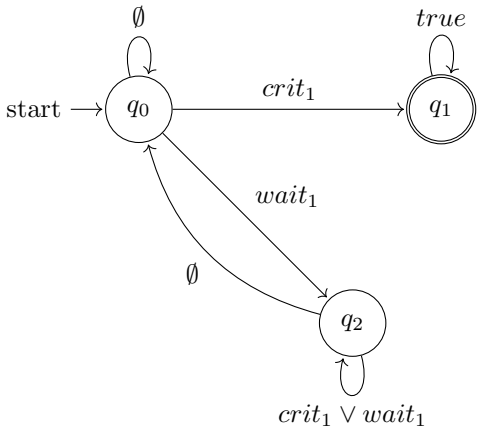


Figure 11: Exercise 4.2 point (b)

(ii) Recall the Semaphore based TS(Figure 16).The product with the previous point's NFA is given in Figure 12, as one can see no state has the right projection equal to q_1.

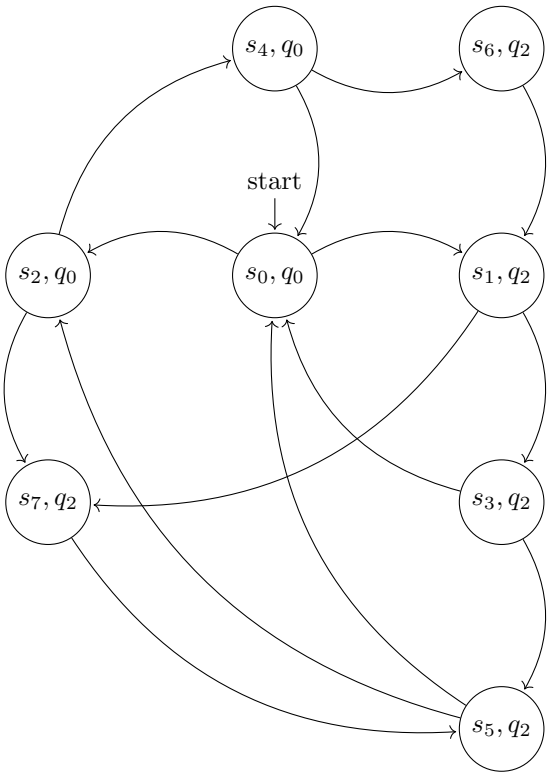


Figure 12: Exercise 4.3. (a) point (ii): Product Graph for Semaphore Algorithm

(b) (i) The NFA can be seen in Figure 13.

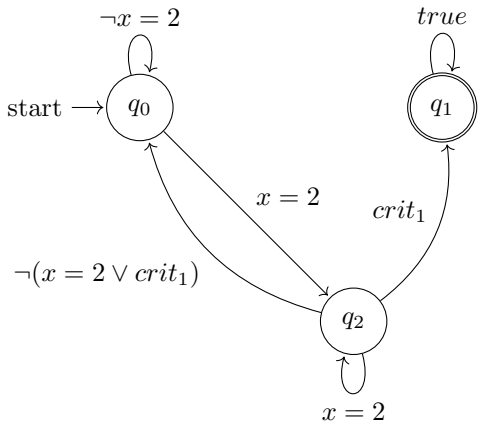


Figure 13: Exercise 4.3 point (b) the NFA of Bad Prefixes for "process 1 never enters its critical section from a state with x=2"

(ii) Recall Peterson's TS in Figure 15 and then we show (part of) the product graph in Figure 14

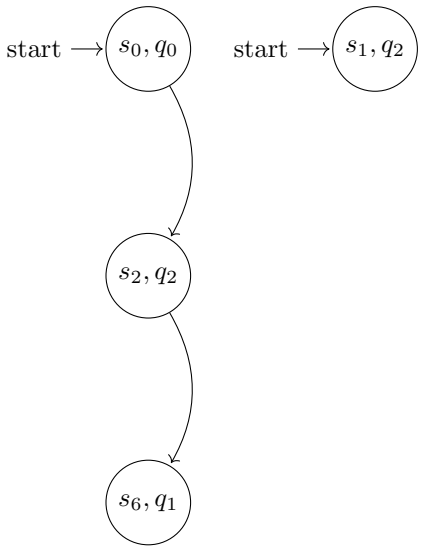


Figure 14: Exercise 4.3 point (b) (ii) part of the product automata for the property "process 1 never enters its critical section from a state with x=2". Clearly we can reach a state s where $L(s) = q_1$. So we conclude the TS does not satisfy the property.

3.0.4 Exercise 4.4

(a) It is true. Take the NFA $N = (Q, \Sigma, \delta, Q_0, F)$ corresponding to $\mathcal{L}$, then by hypothesis this NFA accepts the minimal bad prefixes of P_{safe} and some other bad prefixes. It is enough to remove any outgoing edge from end states in N, obtaining the NFA N', to get an NFA accepting the minimal bad prefixes of P_{safe}. If $w(|w| = n)$ is a minimal bad prefix then it is accepted by N by construction, which means there exists a run π such that $\exists i.\pi_i \in F$ and $\pi_1 \rightarrow^{w_1} \pi_2 \rightarrow^{w_2} ... \rightarrow^{w_n} \pi_i$. Then it is obvious that there cannot be another j such that $j < i$ and $\pi_j \in F$ because this would mean $w' = w_1...w_j$ is a bad prefix (since $L \subseteq BadPref(P_{safe})$) of a minimal bad prefix w. Then we know that N' can imitate this same path and thus accept any minimal bad prefix of P_{safe}. For the other direction, namely that N' accepts only the (minimal) bad prefixes: since N' is N after we remove some edges it cannot expand the language accepted, thus N' accepts only Bad Prefixes of L_{safe} and because no end state has outgoing transition it accepts only the minimal ones.

(b) False. Consider the regular safety property $\emptyset$, then $MinBadPref(P_{safe}) = \emptyset$ and $BadPref(P_{safe}) = (2^{\{0,1\}})^{\omega}$. Now consider $L = \{\{0\}^n\{1\}^n + 1 | \forall n \geq 0\}$ which is clearly not regular.

3.0.5 Exercise 4.5

3.1 Appendix

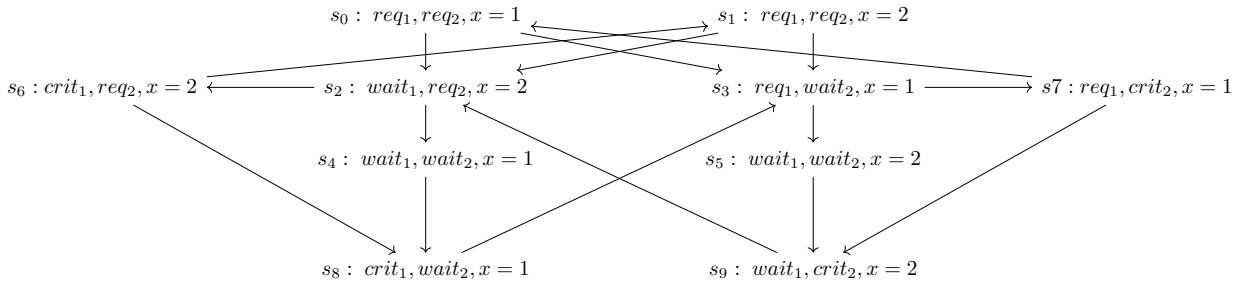


Figure 15: Transition Graph for Peterson Algorithm, TS_{pet}

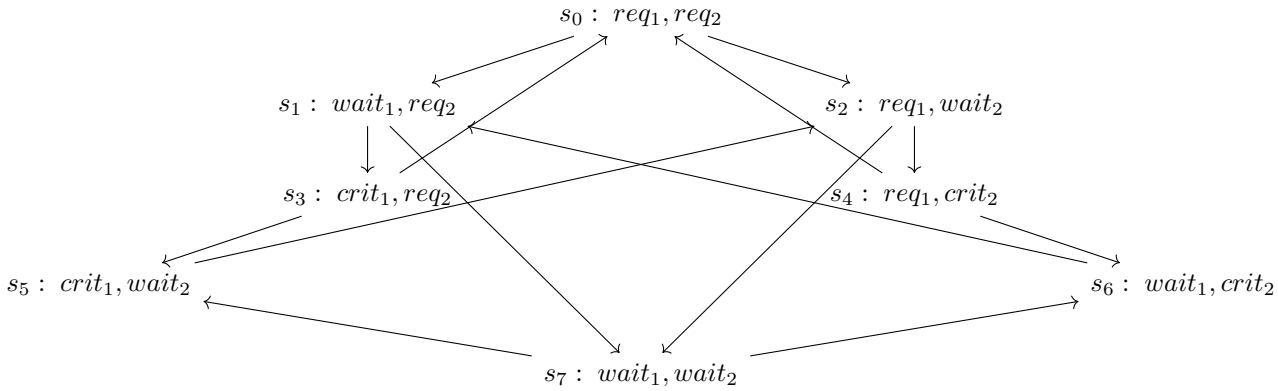


Figure 16: Transition Graph for Semaphore Algorithm, TS_{sem}