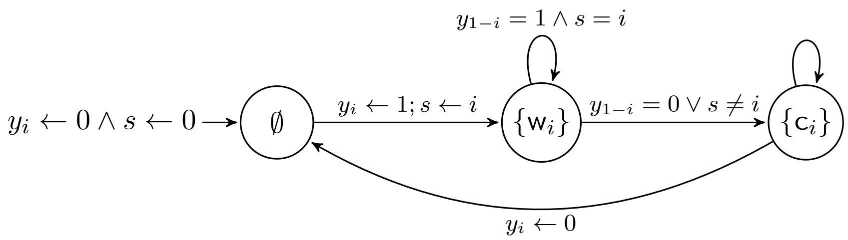
Home Assignment 1: Safety and Liveness (Solutions)

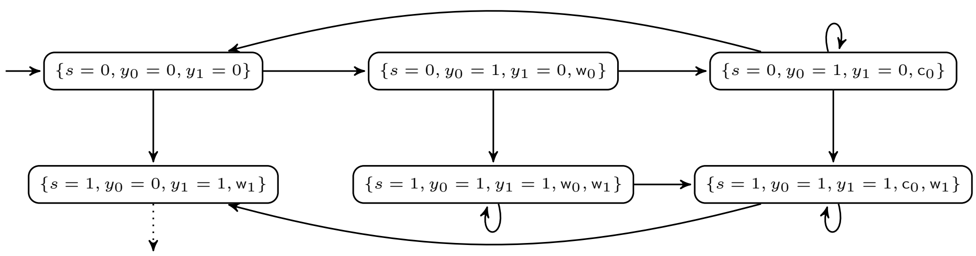
Here are partial solutions and hints for the exercises that caused some difficulties.

Exercise 1 (A Mutual Exclusion Protocol).

1. The construction of the transition system was, quite surprisingly, often incorrect. (A very similar mutual exclusion protocol is detailed in the lecture notes...) Very often, the loops on the waiting and critical section states were missing.



2. The best way to see that mutual exclusion holds is to construct the "low level" transition system and check that no state with both c_0 and c_1 holding at the same time is accessible. Here is half of it:



3. A LTL formula for mutual exclusion is $\mathsf{G}(\neg(\mathsf{c}_0 \wedge \mathsf{c}_1))$.

4. In the transition system above, one can see that the fairness constraint ("the other process does not stay forever in the critical section") takes care of the loops on the states labeled with c_0 or c_1, but not on the states labeled with both w_0 and w_1 (middle state). There is an infinite execution that stays forever in this state, without ever granting the access to the critical section although the processes are waiting. In short, the fairness assumption is not strong enough to guarantee freedom of starvation.

5. A LTL formula for freedom of starvation has to include the fairness constraint: $\bigwedge_{i \in \{0,1\}} (\mathsf{GF}\neg \mathsf{c}_{1-i}) \Rightarrow \mathsf{G}(\mathsf{w}_i \Rightarrow \mathsf{F}\mathsf{c}_i)$, or other reasonable variations.

Exercise 4 (Separation into Past and Future)**.** Aperiodic languages seem to have been a source of difficulties. There are several characterizations of these languages over Σ^∞:

- as the languages defined by LTL($\mathsf{Y}, \mathsf{S}, \mathsf{X}, \mathsf{U}$) or LTL($\mathsf{X}, \mathsf{U}$) formulæ, (this was the definition proposed in the subject),
- as the languages defined by first order FO($<$) formulæ,
- as the languages defined by star-free regular expressions with complement,
- as the languages recognized by morphisms into aperiodic finite structures (monoids or ω-semigroups depending whether we are considering finite or infinite words),
- as the languages defined by counter-free finite or Büchi automata (depending whether we are considering finite or infinite words).

Any of these definitions could be used, depending on your personal taste, but everything could be done using the first.

1. Let $L \subseteq \Sigma^+$ be an aperiodic language of finite words. The exercise was to show that L can be associated with a pure past formula φ such that

$$L = \{w = a_0 a_1 \cdots a_n \in \Sigma^+ \mid w, n \models \varphi\} .$$

The first thing to observe is that aperiodic languages are closed under reversal (aka mirror). Using the LTL($\mathsf{Y}, \mathsf{S}, \mathsf{X}, \mathsf{U}$) characterization of aperiodic languages, this can be proved by exhibiting a formula φ for L, by exchanging past and future modalities (i.e. $\mathsf{X} \leftrightarrow \mathsf{Y}$ and $\mathsf{U} \leftrightarrow \mathsf{S}$) in φ (obtaining a new formula φ'), and adding the constraint that the formula should be evaluated from the end of the string (by considering $\mathsf{F}(\varphi' \wedge \neg \mathsf{X}\top)$—this last formula defines the reversal of L).

Then, by the LTL(X, U) characterization of aperiodic languages, one can find a pure future formula ψ for the *reversal* of L, from which the desired pure past formula φ is obtained by exchanging Y for X and S for U.

2. The purpose of this exercise was to find a decomposition

$$L = \bigcup_{j \in J} P_j \cdot a_j \cdot F_j .$$

Of course, given the next question, not any decomposition would do. We have a simple decomposition:

$$L = \bigcup_{w \in L} w = \bigcup_{uav \in L, u \in \Sigma^*, a \in \Sigma, v \in \Sigma^\omega} uav .$$

Then each such uav is such that

$$[u] \cdot a \cdot [v] \subseteq [u] \cdot [a] \cdot [v] \subseteq [uav]$$

since $a \in [a]$ and $[x] \cdot [y] \subseteq [xy]$ for any x in Σ^* and y in Σ^ω (here we extend $\sim_\mu$ to a relation on Σ^* with $[\varepsilon] = \{\varepsilon\}$, this simplifies matters later). Since furthermore $w \in L$ implies $[w] \subseteq L$, we have

$$\bigcup_{uav \in L, u \in \Sigma^*, a \in \Sigma, v \in \Sigma^\omega} [u] \cdot a \cdot [v] \subseteq L \ .$$

The reverse inclusion holds vacuously since $uav \in [u] \cdot a \cdot [v]$. Hence we have the desired decomposition since $\sim_\mu$ and $\approx_\mu$ are of finite index, Σ is a finite alphabet, and each equivalence class is an aperiodic language.

Let us anticipate the next question and consider

$$P = \bigcup_{uav \in L, u \in \Sigma^*, a \in \Sigma, v \in \Sigma^\omega} [u] \cdot a \ .$$

That $\mathsf{Pref}(L) \backslash \{\varepsilon\} \subseteq P$ is again obvious. The converse inclusion holds because any word in any $[u] \cdot a$ can be completed with any word in $[v]$ into a word of L (again $[u] \cdot a \cdot [v] \subseteq [uav] \subseteq L$).

3. The separation theorem and how to prove it were mostly well understood, but the details were usually not quite right. The best was probably to define the separation formula as

$$\varphi = \bigvee_{j \in J, P_j \neq \{\varepsilon\}} \mathsf{Y}\varphi_j \wedge a_j \wedge \mathsf{X}\varphi'_j \vee \bigvee_{j \in J, P_j = \{\varepsilon\}} (\neg \mathsf{Y} \top) \wedge a_j \wedge \mathsf{X}\varphi'_j$$

with φ_j the LTL(Y, S) formula associated with $P_j \neq \{\varepsilon\}$ (thanks to Question 4.1) and φ'_j the LTL(X, U) formula associated with F_j.

Exercise 6 (Characteristic Liveness Formulæ)**.**

1. The characteristic liveness formula for the starvation freedom property was seldom correct. Here is one for process P_0; we set $\varphi_0 = \mathsf{G}(\mathsf{w}_0 \Rightarrow \mathsf{Fc}_0)$):

$$\begin{aligned}\mathsf{F}\Big(&\bigvee_{a \in \Sigma} \top \wedge a \wedge \mathsf{XGc}_1\\ &\vee(((\neg \mathsf{w}_0)\ \mathsf{S}\ \mathsf{c}_0) \vee \neg(\top\ \mathsf{S}\ \mathsf{w}_0)) \wedge a \wedge \varphi_0\\ &\vee(\neg \mathsf{c}_0\ \mathsf{S}\ \mathsf{w}_0) \wedge a \wedge \mathsf{Fc}_0 \wedge \varphi_0\Big) \ .\end{aligned}$$

One can easily check that the disjunction of past parts forms a valid formula—although some conjuncts might not be satisfiable, e.g. $(\neg \mathsf{c}_0\ \mathsf{S}\ \mathsf{w}_0) \wedge a$ if $\mathsf{c}_0 \in a$.

Exercise 7 (Model Checking Safety Formulæ).

4. We want to prove that the model checking problem for finite Kripke structures and characteristic safety formulæ is PSPACE-complete. By the previous question, you should have an algorithm in polynomial space for this problem, and thus the remaining issue is to prove PSPACE-hardness.

   The reduction from QBF given in the lecture notes is easy to adapt for this purpose: given a QBF instance $\gamma = Q_1 x_1 \cdots Q_n x_n \bigwedge_{1 \leq i \leq m} \bigvee_{1 \leq j \leq k_i} a_{ij}$ with each Q_l a quantifier in $\{\forall, \exists\}$ and each a_{ij} a literal of form x_l or $\neg x_l$ for some $1 \leq l \leq m$, construct the same Kripke structure and the formulæ

$$\psi = \bigwedge_{1 \leq l \leq m} \left(s_l \Rightarrow \left((\neg e_l \wedge \bigwedge_{a_{ij} = x_l} \neg a_{ij}) \mathsf{~S~} \mathsf{Y} x_l^f \right) \vee \left((\neg e_l \wedge \bigwedge_{a_{ij} = \neg x_l} \neg a_{ij}) \mathsf{~S~} \mathsf{Y} x_l^t \right) \vee \neg (\top \mathsf{~S~} e_l) \right)$$

$$\varphi = \bigwedge_{l \mid Q_l = \forall} \Big(s_{l-1} \Rightarrow ((\neg e_{l-1} \mathsf{~S~} x_l^t) \wedge (\neg e_{l-1} \mathsf{~S~} x_l^f)) \vee \neg (\top \mathsf{~S~} e_l) \Big)$$

   Then, γ is valid iff the system verifies existentially $\mathsf{G}(\psi \wedge \varphi)$. The runs that verify this formula are indeed the same as the ones in the lecture notes.